RICE UNIVERSITY

New Approaches to Modeling Multi-Port Scattering Parameters

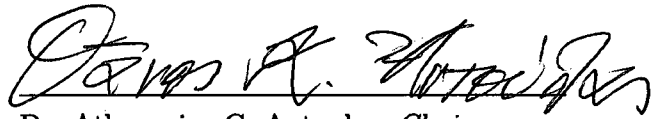
by

Sanda Lefteriu

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE

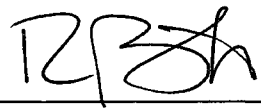
Master of Science

APPROVED, THESIS COMMITTEE:



Dr. Athanasios C. Antoulas, *Chair*
Professor
Electrical and Computer Engineering

Dr. Mark Embree
Associate Professor
Computational and Applied Mathematics



Dr. Richard G. Baraniuk
Victor E. Cameron Professor
Electrical and Computer Engineering

HOUSTON, TEXAS

JUNE 2008

ABSTRACT

New Approaches to Modeling Multi-Port Scattering Parameters

by

Sanda Lefteriu

This work addresses the problem of building a macromodel from frequency response measurements by means of a stable and passive linear dynamical system in state-space representation. The proposed algorithms are based on a system-theoretic tool, the matrix pencil of the shifted Loewner and Loewner matrices. Their performance is compared with that of the widely-used vector fitting in terms of the computational time required to build such a model and the accuracy of the interpolating system, when the same order model is constructed, and it is shown that our algorithms render better models in less time.

Even though the main application we have in mind is modeling the scattering parameters of an electromagnetic device, no modifications are needed when the admittance parameters are provided instead. Last, our algorithms are especially suited for devices with a large number of ports because the data matrices are collapsed into vectors.

Acknowledgements

Foremost, I would like to thank my advisor, Dr. Athanasios Antoulas, for all his support and guidance, starting with the internship I conducted under his supervision in summer 2005 and continuing with the Guided Research project during spring 2006. I hereby express my deep and sincere gratitude for allowing me to become one of his students here at Rice. It is indeed wonderful to work in an area I enjoy a lot, supervised by one of the experts in the field.

Rice University is a great place to be and this is mainly due to the people. I would like to thank the members of my committee, Dr. Mark Embree and Dr. Richard Baraniuk, as well as Dr. Danny Sorensen, Petros Boufounos and all the other professors and students who contributed to my knowledge and better understanding of concepts which are not in my subject area at the first glance, but will definitely be useful in the near future.

I left the special thanks for the end, but it was intentionally because they are the most important people in my life: my family back home in Romania, who are always sad every time I have to leave the country for another six months. And here I also include our dog. I owe everything to my mother, my father and my sister. Last, but not least, I would like to thank Jacob for listening and giving advice.

Contents

List of Figures

List of Tables

List of Algorithms

CHAPTER 1

Introduction and Motivation

Demand for high data bandwidth requires accurate simulation of entire complex systems such as chips, packages or boards. In particular, electromagnetic effects in passive electronic structures with arbitrarily shaped layouts need to be simulated. There are several ways of approaching this problem. The first is to discretize Maxwell's equations on the domains of interest, leading to an internal representation in descriptor-form which is of extremely high order (the number of unknows depends on the size of the discretization grid, but typical values are in the range of thousands). Therefore, model reduction methods need to be applied to reduce the dimension to a manageable size. A second approach is to measure the frequency response of the device over a frequency band of interest. In the case of interconnect structures, one usually models them as RLC structures. However, when such a model is not available, the *admittance* or *scattering* parameters are measured or simulated. Afterwards, a passive macromodel of low complexity should be constructed such that it is consistent with the data. The problem of constructing a system which approximates given frequency response measurements at various frequencies is known as the *rational interpolation problem* and has been studied thoroughly (see [1] for a survey).

Several techniques have been developped in the electronics community. Most of

the algorithms proposed are based on least-squares approximations, for example [2], but, due to ill-conditioning, their application is restricted to narrow frequency bands and small orders of the model. Other approaches are moment-based, like [3], but, as pointed out in [4], even though the underlying theory is sound, this method involves three steps which are challenging from the numerical point of view: accurately solve for the Y-parameters in the time domain with IFFT, use numerical intergration to compute the moments of the admittance parameters and last, use a Padé approximation to find the poles and residues from these moments. One other popular approach is frequency domain subspace identification [5], but, based on the experiments performed in [6], we conclude that the method fails quite often or requires a large computational time. Nevertheless, the most popular algorithm is the well-known vector fitting method [7], [8], [9], which has lately become the industry standard.

The methods we are be proposing in this work are all developed under the Loewner framework. They are fast and accurate algorithms, which construct models of low complexity and are especially designed for structures with a large number of ports.

Our algorithms are general and can be applied to any kind of frequency-domain data fitting, but we are mainly interested in using them to model scattering parameters (S-parameters). When the scattering parameters of an electromagnetic device are provided, the interpolating system must satisfy additional constraints, like stability and bounded-realness (passivity). When the admittance parameters are to be interpolated, the model should be stable and positive-real.

Current applications in electromagnetic devices require a model of reasonable dimensions which approximates the response accurately in the desired frequency range. Our algorithms aim to address this issue, especially in the cases where these devices have a large number of ports (e.g. field programmable gate arrays), for which currently available vector fitting, as well as the other methods, proves to be prohibitively expensive. The computational complexity of column-wise vector fitting scales with

the fourth power of the number of ports N_p, so for devices with hundreds of ports, this becomes unfeasible. Some of the algorithms we propose, however, scale with the second or third power of the number of ports N_p, yielding better models than vector fitting in less time.

We use a black-box approach by not assuming any underlying stucture of the systems to be modeled (reciprocity of the network is not assumed, meaning that we can deal with non-symmetric, as well as symmetric, S-parameter matrices). The advantage of this approach is that a system can be modeled independently of the knowledge of its internal logic [10]. Moreover, our algorithms construct the models from the available measured data by arranging it in a clever way, so, except for the data file and the desired order or accuracy of the interpolant, no other user input is required. Vector fitting, on the other hand, requires a set of good starting poles for the pole-relocation process to be successful and produce good macromodels.

This thesis is organized as follows. Chapter 2 defines the terms we will be using throughout this work. Chapter 3 presents the basic ideas behind the widely-used vector fitting algorithm. Chapter 4 presents the theoretical foundation upon which our algorithms are based, namely the Loewner matrix pencil. Several implementations have been described in chapter 5 and their computational complexity has been investigated in chapter 6. The algorithms have been compared to vector fitting in chapter 7, in terms of performance, using a representative set of examples, both theoretical and obtained from measuremtents. Chapter 8 concludes this work. The pseudocode of the algorithms is included in appendix A. We have tested our algorithms extensively and further examples are analyzed in appendix B.

CHAPTER 2

Definitions

2.1 Electromagnetics

Definition 2.1.1. Scattering Parameters, *or* S-parameters, *are the reflection and transmission coefficients between incident and reflection waves. They completely describe the behavior of a linear device. For each frequency, each parameter is a complex number and is typically characterized by magnitude and phase.*

The S-parameter matrix for the 2-port network is the most common and it serves as the building block for generating the higher order matrices of larger networks. In the 2-port case the relationship between the reflected, the incident power waves and the S-parameter matrix is given by:

$$\begin{bmatrix} b_1(s) \\ b_2(s) \end{bmatrix} = \begin{bmatrix} S_{11}(s) & S_{12}(s) \\ S_{21}(s) & S_{22}(s) \end{bmatrix} \begin{bmatrix} a_1(s) \\ a_2(s) \end{bmatrix}. \tag{2.1}$$

The S-parameters in this case have the following interpretation:

- $S_{11}(s)$ is the input port voltage reflection coefficient
- $S_{12}(s)$ is the reverse voltage gain

- $S_{21}(s)$ is the forward voltage gain
- $S_{22}(s)$ is the output port voltage reflection coefficient.

A compact way of writing the relationship 2.1 for an arbitrary number of ports N_p is

$$\mathbf{b} = \mathbf{S}\mathbf{a}$$

where

$$\mathbf{b} = \begin{bmatrix} b_1(s) \\ \vdots \\ b_{N_p}(s) \end{bmatrix} \tag{2.2}$$

$$\mathbf{S} = \begin{bmatrix} S_{1,1} & \cdots & S_{1,N_p} \\ \vdots & \ddots & \vdots \\ S_{N_p,1} & \cdots & S_{N_p,N_p} \end{bmatrix} \tag{2.3}$$

$$\mathbf{a} = \begin{bmatrix} a_1(s) \\ \vdots \\ a_{N_p}(s) \end{bmatrix} \tag{2.4}$$

Assuming that the reference impedance Z_0 is the same for all ports (usually taken as $Z_0 = 50\Omega$), the incident and reflected waves may be expressed in terms of the voltages ($\mathbf{V}$) and currents ($\mathbf{I}$) at each port as:

$$\mathbf{b} = \frac{1}{2}\frac{\mathbf{V} - Z_0^*\mathbf{I}}{\sqrt{|Re(Z_0)|}} \tag{2.5}$$

$$\mathbf{a} = \frac{1}{2}\frac{\mathbf{V} + Z_0\mathbf{I}}{\sqrt{|Re(Z_0)|}} \tag{2.6}$$

Definition 2.1.2. *A network is* reciprocal *if it is passive and contains only isotropic materials that influence the transmitted signal.*

For example, attenuators, cables, splitters and combiners are all reciprocal networks. In this case, the S-parameter matrix will be symmetric: $S_{mn} = S_{nm}$ or, for a 2-port network, $S_{12} = S_{21}$. All networks which include anisotropic materials in the transmission medium such as those containing ferrite components will be non-reciprocal.

Definition 2.1.3. *A* loss-free network *is one which does not dissipate any power, or the sum of the incident powers at all ports is equal to the sum of the reflected powers at all ports:* $\sum_{i=1}^{N_p} |a_i|^2 = \sum_{i=1}^{N_p} |b_i|^2$, *where* N_p *stands for the number of ports.*

This implies that the S-parameter matrix is unitary for all frequencies $s = j\omega$, or $\mathbf{S}\mathbf{S}^* - \mathbf{I}_{N_p} = \mathbf{0}$, where $(\cdot)^*$ denotes transposition followed by complex conjugation and $\mathbf{I}_{N_p}$ is the identity matrix of dimension $N_p \times N_p$. If the singular value decomposition of $\mathbf{S}$ is $\mathbf{U}\Sigma\mathbf{V}^*$, with $\mathbf{U}$ and $\mathbf{V}$ unitary and Σ a diagonal matrix with the singular values on the diagonal, then

$$
\begin{aligned}
\mathbf{S}\mathbf{S}^* - \mathbf{I}_{N_p} = \mathbf{0} \quad &\Rightarrow \\
(\mathbf{U}\Sigma\mathbf{V}^*)(\mathbf{V}\Sigma\mathbf{U}^*) - \mathbf{I}_{N_p} = \mathbf{0} \quad &\Rightarrow \\
\mathbf{U}\Sigma^2\mathbf{U}^* - \mathbf{U}\mathbf{U}^* = \mathbf{0} \quad &\Rightarrow \\
\mathbf{U}^* \left| \mathbf{U}\left(\Sigma^2 - \mathbf{I}_{N_p}\right)\mathbf{U}^* = \mathbf{0} \right| \mathbf{U} \quad &\Rightarrow \\
\Sigma^2 = \begin{bmatrix} \sigma_1^2 & & \\ & \ddots & \\ & & \sigma_{N_p}^2 \end{bmatrix} = \begin{bmatrix} 1 & & \\ & \ddots & \\ & & 1 \end{bmatrix}.
\end{aligned}
$$

We conclude that, for a loss-free network, the singular values of the S-parameter matrix are all equal to 1.

Definition 2.1.4. *A* lossy network *is one in which the sum of the incident powers at all ports is greater than the sum of the reflected powers at all ports. This is equivalent to saying that the network dissipates power, or:* $\sum_{i=1}^{N_p} |a_i|^2 > \sum_{i=1}^{N_p} |b_i|^2$.

In this case $\sum_{i=1}^{N_p} |a_i|^2 > \sum_{i=1}^{N_p} |b_i|^2$ and $\mathbf{I}-\mathbf{SS}^* > \mathbf{0}$. This implies that the singular values of the S-parameter matrix are stricly smaller than or equal to one:

$$\begin{aligned}
\mathbf{I}-\mathbf{SS}^* > \mathbf{0} &\Rightarrow \\
\mathbf{I}-(\mathbf{U}\Sigma\mathbf{V}^*)(\mathbf{V}\Sigma\mathbf{U}^*) > \mathbf{0} &\Rightarrow \\
\mathbf{U}\mathbf{U}^*-\mathbf{U}\Sigma^2\mathbf{U}^* > \mathbf{0} &\Rightarrow \\
\mathbf{U}^*\left|\mathbf{U}\left(\mathbf{I}-\Sigma^2\right)\mathbf{U}^* > \mathbf{0}\right|\mathbf{U} &\Rightarrow \\
\mathbf{I}-\Sigma^2 > \mathbf{0} \Rightarrow \begin{bmatrix} 1-\sigma_1^2 & & \\ & \ddots & \\ & & 1-\sigma_{N_p}^2 \end{bmatrix} > 0 &\Rightarrow \\
\sigma_1,\ldots,\sigma_{N_p} \leq 1. &
\end{aligned}$$

2.2 System theory

Definition 2.2.1. *A linear dynamical system* Σ *with m-input ports, p-output ports and n-internal variables in* descriptor-form representation *is given by a set of differential and algebraic equations*

$$\Sigma: \quad \mathbf{E}\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t), \quad \mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t), \tag{2.7}$$

where $\mathbf{x}(t)$ *is an internal variable (the state, if* $\mathbf{E}$ *is invertible),* $\mathbf{u}(t)$ *is an input,* $\mathbf{y}(t)$ *is an output, while* $\mathbf{E},\mathbf{A} \in \mathbb{R}^{n\times n}$, $\mathbf{B} \in \mathbb{R}^{n\times m}$, $\mathbf{C} \in \mathbb{R}^{p\times n}$, $\mathbf{D} \in \mathbb{R}^{p\times m}$ *are constant matrices with* $\mathbf{E}$ *possibly singular.*

Definition 2.2.2. *The* transfer function *of* Σ *is*

$$\mathbf{H}(s) = \mathbf{C}\left(s\mathbf{E}-\mathbf{A}\right)^{-1}\mathbf{B}+\mathbf{D}.$$

Definition 2.2.3. *The set of matrices* $[\mathbf{E},\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}]$ *is called a* realization *of* $\mathbf{H}(s)$.

Note that the realization of a transfer function is not unique. The realization of the smallest possible order n is called a minimal realization.

Definition 2.2.4. *The* poles *of a system* Σ *are given by the eigenvalues of the matrix pencil* $(\mathbf{A}, \mathbf{E})$*:*

$$\textit{poles of } \Sigma = \lambda(\mathbf{A}, \mathbf{E}). \tag{2.8}$$

Definition 2.2.5. *A system* Σ *is called* stable *if all its poles are in the right-half plane:*

$$\Sigma \textit{ stable} \Leftrightarrow Re(\lambda(\mathbf{A}, \mathbf{E})) < 0. \tag{2.9}$$

In the general case (whether we are dealing with a lossy or a loss-free network), the singular values of the S-parameter matrix are smaller than or equal to 1. If a linear dynamical system which interpolates the data were to be constructed, it would have to be *bounded real* or *passive.*

Definition 2.2.6. *A* bounded-real *system* Σ *has the same number of inputs and outputs* $(m = p = N_p)$ *and satisfies the following conditions:*

$$\begin{cases} \overline{\mathbf{H}(s)} = \mathbf{H}(\overline{s}), \textit{ for all } s \in \mathbb{C} \\ \mathbf{H}(s) \textit{ is analytic for } Re(s) > 0, \textit{(i.e., } \mathbf{H} \textit{ has no poles} \\ \qquad \textit{in the right-half of the complex plane)} \\ \mathbf{I} - \mathbf{H}(s)\mathbf{H}^*(-s) \geq 0 \textit{ for } Re(s) > 0 \end{cases}$$

where $\overline{(\cdot)}$ *denotes complex conjugation. Bounded-realness is equivalent to the* $\mathcal{H}_\infty$*-norm of the transfer function being less than or equal to one:*

$$\|\mathbf{H}\|_{\mathcal{H}_\infty} = sup_\omega \|\mathbf{H}(j\omega)\| = \sigma_1(\mathbf{H}(j\omega)) \leq 1, \tag{2.10}$$

where $\sigma_1(\cdot)$ *denotes the largest singular value of the matrix* $(\cdot)$*. For details on these issues we refer to [1].*

In case of a system with only one input port and one output port, the $\mathcal{H}_\infty$-norm is the peak on the freguency response. If the system has several inputs and outputs ($N_p > 1$), the $\mathcal{H}_\infty$-norm is given by the maximum of the largest singular value of the $N_p \times N_p$ transfer function matrix $\mathbf{H}$, as a function of frequency.

We will say that the system $\mathbf{\Sigma}$ models the data set obtained from measuring the scattering coefficients of an electromagnetic device with $N_p = m = p$ number of ports for a number of N_s frequency samples

$$\left(j\omega_i, \mathbf{S}_i := \begin{bmatrix} S_{11,i} & \cdots & S_{1N_p,i} \\ \vdots & \vdots & \vdots \\ S_{N_p1,i} & \cdots & S_{N_pN_p,i} \end{bmatrix} \right), i = 1, \cdots, N_s$$

if the value of the transfer function evaluated at the sampling point $j\omega_i$ is close to the scattering matrix at the frequency ω_i :

$$\mathbf{H}(j\omega_i) \approx \mathbf{S}_i, \ i = 1, \ldots, N_s.$$

At this point, let us define an error quantity called the error matrix at a specific frequency which is computed as the difference between the transfer function evaluated at that frequency and the measured S-parameters:

$$\mathbf{H}(j\omega_i) - \mathbf{S}_i = Err(j\omega_i), \ i = 1, \ldots, N_s. \tag{2.11}$$

Clearly, if the norm of all the N_s error matrices is small (take for example the $\mathcal{H}_\infty$ norm, which is given by the largest singular value), then our model is accurate. Otherwise, if the errors are large, the model is poor.

CHAPTER 3

Vector Fitting

In this chapter we will provide a short review of the basic ideas behind vector fitting (VF) [7], [8], [10], [11], [6].

Let us consider the case of modeling a SISO-system. We aim at finding the rational function approximation of its transfer function

$$H(s) \approx f(s),\ f(s) = \sum_{i=1}^{k} \frac{c_i}{s - a_i} + d + se. \tag{3.1}$$

The finite residues c_i and poles a_i are either real quantities or come in complex conjugate pairs, while d and e are real. The problem is to estimate all coefficients in (3.1) so that a least squares approximation of $H(s)$ is obtained over a given frequency interval. As it is, this is a nonlinear problem, but vector fitting solves it sequentially as a linear problem in two stages, both times with known poles.

Stage 1: pole indentification

Specify a set of starting poles $\bar{a}_i$ for an unknown weighting function $\sigma(s)$ and multiply it with $f(s)$. This gives the augmented problem:

$$\begin{bmatrix} \sigma(s)f(s) \\ \sigma(s) \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{k} \frac{c_i}{s-\bar{a}_i} + d + se \\ \sum_{i=1}^{k} \frac{\tilde{c}_i}{s-\bar{a}_i} + 1 \end{bmatrix}. \tag{3.2}$$

The rational approximation for $\sigma(s)$ has the same poles as the approximation for $\sigma(s)f(s)$. Also, $\sigma(s)$ is forced to approach unity at high frequencies. Multiplying the second row in (3.2) by $f(s)$ yields:

$$\sum_{i=1}^{k} \frac{c_i}{s-\bar{a}_i} + d + se = \left(\sum_{i=1}^{k} \frac{\tilde{c}_i}{s-\bar{a}_i} + 1 \right) f(s) \tag{3.3}$$

or

$$(\sigma f)_{fit}(s) = \sigma_{fit}(s) f(s) \tag{3.4}$$

Equation (3.3) is linear in its unknowns c_i, d, e, $\tilde{c}_i$. We use the fact that $f(s)$ interpolates the original transfer function $H(s)$ at the frequency samples. Writing (3.3) for the frequency points at which we know the value of f gives the overdetermined linear problem $\mathbf{Ax} = \mathbf{b}$ where the unknowns are in the solution vector $\mathbf{x}$. This is solved as a least squares problem.

$$\mathbf{A} = \begin{bmatrix} \frac{1}{s_1-\bar{a}_1} & \cdots & \frac{1}{s_1-\bar{a}_n} & 1 & s_1 & \frac{-S_1}{s_1-\bar{a}_1} & \cdots & \frac{-S_1}{s_1-\bar{a}_n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{1}{s_{N_s}-\bar{a}_1} & \cdots & \frac{1}{s_{N_s}-\bar{a}_n} & 1 & s_{N_s} & \frac{-S_{N_s}}{s_{N_s}-\bar{a}_1} & \cdots & \frac{-S_{N_s}}{s_{N_s}-\bar{a}_n} \end{bmatrix} \in \mathbb{C}^{N_s \times (2n+2)}, \tag{3.5}$$

$$\mathbf{x} = \begin{bmatrix} c_1 & \cdots & c_n & d & e & \tilde{c}_1 & \cdots & \tilde{c}_n \end{bmatrix}^T \in \mathbb{C}^{(2n+2)\times 1}, \tag{3.6}$$

$$\mathbf{b} = \begin{bmatrix} S_1 & \cdots & S_{N_s} \end{bmatrix}^T \in \mathbb{C}^{N_s \times 1} \tag{3.7}$$

where $s_i = j\omega_i$ and $S_i = H(s_i)$, for $i = 1, \ldots, N_s$.

When we fit a vector-valued function for which we have some measurements $\mathbf{M}_i \in \mathbb{C}^{N_p \times 1}$, for $i = 1 \ldots, N_s$, available, the linear system presented in equations (3.5)-(3.7)

changes to:

$$\mathbf{A} = \begin{bmatrix} \frac{\mathbf{I}_{N_p}}{s_1-\bar{a}_1} & \cdots & \frac{\mathbf{I}_{N_p}}{s_1-\bar{a}_n} & \mathbf{I}_{N_p} & s_1\mathbf{I}_{N_p} & \frac{-\mathbf{M}_1}{s_1-\bar{a}_1} & \cdots & \frac{-\mathbf{M}_1}{s_1-\bar{a}_n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\mathbf{I}_{N_p}}{s_{N_s}-\bar{a}_1} & \cdots & \frac{\mathbf{I}_{N_p}}{s_{N_s}-\bar{a}_n} & \mathbf{I}_{N_p} & s_{N_s}\mathbf{I}_{N_p} & \frac{-\mathbf{M}_{N_s}}{s_{N_s}-\bar{a}_1} & \cdots & \frac{-\mathbf{M}_{N_s}}{s_{N_s}-\bar{a}_n} \end{bmatrix}, \tag{3.8}$$

$$\mathbf{x} = \begin{bmatrix} \mathbf{c}_1 & \cdots & \mathbf{c}_n & \mathbf{d} & \mathbf{e} & \tilde{\mathbf{c}}_1 & \cdots & \tilde{\mathbf{c}}_n \end{bmatrix}^T, \tag{3.9}$$

$$\mathbf{b} = \begin{bmatrix} \mathbf{M}_1 & \cdots & \mathbf{M}_{N_s} \end{bmatrix}^T \tag{3.10}$$

where $s_i = j\omega_i$, $\mathbf{M}_i = \mathbf{H}(s_i)$, for $i = 1, \ldots, N_s$, and $\mathbf{I}_{N_p}$ is the identity matrix of dimension N_p. The dimensions of the matrices in Eq.(3.8)-(3.10) are $\mathbf{A} \in \mathbb{C}^{(N_s N_p)\times(n(N_p+1)+2N_p)}$, $\mathbf{x} \in \mathbb{C}^{(n(N_p+1)+2N_p)\times 1}$ and $\mathbf{b} \in \mathbb{C}^{(N_s N_p)\times 1}$.

Rewriting Eq. (3.4) in the pole-zero expression, we obtain:

$$(\sigma f)_{fit}(s) = e\frac{\prod_{i=1}^{k}(s-z_i)}{\prod_{i=1}^{k}(s-\bar{a}_i)} \tag{3.11}$$

$$\sigma_{fit}(s) = \frac{\prod_{i=1}^{k}(s-\bar{z}_i)}{\prod_{i=1}^{k}(s-\bar{a}_i)} \tag{3.12}$$

so $f(s)$ can be computed as

$$f(s) = \frac{(\sigma f)_{fit}(s)}{\sigma_{fit}(s)} = e\frac{\prod_{i=1}^{k}(s-z_i)}{\prod_{i=1}^{k}(s-\bar{z}_i)}. \tag{3.13}$$

This shows that the zeros of the weighting function $\sigma_{fit}(s)$ become the poles of $f(s)$. The zeros $\bar{z}_i$ are found by solving an eigenvalue problem. Now we iterate, using the new poles $\bar{z}_i$ as starting poles. By the end of the iteration process, we would have found the poles of $f(s)$, namely a_i. This is a pole relocation process. The next stage deals with finding the residues c_i, as well as the asymptotic terms d and e.

Stage 2: residue indentification

The first idea would be to solve for the residues c_i, as well as the asymptotic terms d and e, from Eq. (3.13). However, this is not very computationally reliable, so one should solve the original problem (3.1) using the already determined poles of $f(s)$. This leads to another overdetermined linear system which is solved as a least squares problem with

$$\mathbf{A} = \begin{bmatrix} \frac{1}{s_1-\bar{a}_1} & \cdots & \frac{1}{s_1-\bar{a}_n} & 1 & s_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{1}{s_{N_s}-\bar{a}_1} & \cdots & \frac{1}{s_{N_s}-\bar{a}_n} & 1 & s_{N_s} \end{bmatrix} \in \mathbb{C}^{N_s \times (n+2)}, \tag{3.14}$$

$$\mathbf{x} = \begin{bmatrix} c_1 & \cdots & c_n & d & e \end{bmatrix}^T \in \mathbb{C}^{(n+2)\times 1}, \tag{3.15}$$

$$\mathbf{b} = \begin{bmatrix} S_1 & \cdots & S_{N_s} \end{bmatrix}^T \in \mathbb{C}^{N_s \times 1} \tag{3.16}$$

When we fit a vector-valued function for which we have some measurements $\mathbf{M}_i \in \mathbb{C}^{N_p \times 1}$, for $i = 1 \ldots, N_s$, available, the linear system presented in equations (3.14)-(3.16) changes to:

$$\mathbf{A} = \begin{bmatrix} \frac{\mathbf{I}_{N_p}}{s_1-\bar{a}_1} & \cdots & \frac{\mathbf{I}_{N_p}}{s_1-\bar{a}_n} & 1 & s_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\mathbf{I}_{N_p}}{s_{N_s}-\bar{a}_1} & \cdots & \frac{\mathbf{I}_{N_p}}{s_{N_s}-\bar{a}_n} & 1 & s_{N_s} \end{bmatrix} \in \mathbb{C}^{(N_s N_p)\times((n+2)N_p)}, \tag{3.17}$$

$$\mathbf{x} = \begin{bmatrix} \mathbf{c}_1 & \cdots & \mathbf{c}_n & \mathbf{d} & \mathbf{e} \end{bmatrix}^T \in \mathbb{C}^{((n+2)N_p)\times 1}, \tag{3.18}$$

$$\mathbf{b} = \begin{bmatrix} \mathbf{M}_1 & \cdots & \mathbf{M}_{N_s} \end{bmatrix}^T \in \mathbb{C}^{(N_s N_p)\times 1}. \tag{3.19}$$

The theory presented above assumes a single-input single-output or a single-input multiple-output system. In the case of multiple-input multiple-output systems, fitting can be performed matrix-wise, column-wise or element-wise. Since vector fitting works for scalar or vector-valued rational functions, matrix-wise fitting stacks all the

columns into a column vector $vec(\mathbf{H}(s))$ and fits this with the samples $vec(\mathbf{S})$. As detailed in [6], for matrix-wise fitting, all entries of the transfer function will have the same poles and the resulting state-space realization is of dimension $N \cdot N_p$, where N is the number of starting poles, when no d or e are required. For column-wise fitting, each column of the transfer function will be fitted by a common set of poles and the resulting state-space realization is, again, of dimension $N \cdot N_p$, with N as the number of starting poles (the same starting poles are used for all columns), when no d or e are required. Last, for element-wise fitting, different sets of poles will be obtained for each entry of the transfer function, and the resulting state-space realization is of dimension $N \cdot N_p^2$, with N the number of starting poles (the same starting poles are used for all entries), when no d or e are required.

For all the numerical examples presented in Chapter 7, vector fitting was used to fit MIMO systems with column-wise fitting. Even though [6] has shown experimentally that element-wise fitting is faster and yields smaller errors, the order of the resulting models will be too large, as they are multiples of the second power of the number of ports N_p. Therefore, if a data set comes from a device with $N_p = 50$ ports, the smallest dimension that a resulting model built with VF can have is $N_p^2 = 2500$. The main goal in model reduction is constructing models of the least possible order which approximate the frequency response in a certain frequency band. Since models of large dimensions are expensive to use in future simulations, we decided to use column-wise fitting instead of element-wise fitting.

Remark. As pointed out in [11], there is no proof on when or how fast vector fitting converges. Additionally, the convergence properties of the pole relocation process strongly depend on the starting poles. Experimentally, it has been shown in [7] that the starting poles should be chosen as either real numbers linearly distributed in the desired frequency range, or complex conjugate pairs with the imaginary part linearly distributed in the desired frequency range and the real part as 1% of the

imaginary part. Moreover, unstable poles may result during the iterative process due to the fact that, at iteration step l, the poles of the approximating rational function are the zeroes computed at iteration step $l-1$ and the zeroes of a function may be anywhere in the complex plane. To avoid the instability of the resulting models, it was proposed in [7] to either remove the unstable poles or flip the sign of their real part to move them in the left-half plane.

Even though vector fitting generates stable models of the frequency data, the issue of passivity still needs to be addressed. In case the $\mathcal{H}_\infty$-norm of the macromodel is slightly above one (i.e. by 10% or less), the easiest way to obtain a passive model is to divide either the **B** or the **C** state-space matrix of the corresponding realization by that maximum. Another post-processing option for ensuring passivity would be to apply the passivity enforcement algorithm described in [12]. This approach was used in [13], where non-passive models generated by vector fitting were made passive by perturbing the purely imaginary eigenvalues of the associated Hamiltonian matrix. It is well known that purely imaginary eigenvalues of the associated Hamiltonian matrix of a system correspond to frequencies where the plot of the singular values of the system crosses the 0dB line. The eigenvalues of the associated Hamiltonian matrix (for the state-space representation) or Hamiltonian pencil (for the descriptor-form representation) are also called the *spectral zeros of the system.* Assuming a stable matrix rational function obtained with vector fitting, several optimization methods [14], [15], [16] using the positive-real lemma have been proposed as aposteriori passivity enforcement methods. They compute the passive model which minimizes the error between the original data and the model's frequency response, keeping the starting poles unchanged. On the other hand, for admittance Y-parameters, the passivity constraint was incorporated into the vector fitting algorithm [17].

CHAPTER 4

Theoretical Aspects

4.1 Review of existing results

4.1.1 Tangential interpolation

Tangential interpolation is the most general form of the interpolation problem, where the matrix data are sampled directionally on the left and on the right. In this case, the data are split into the *right interpolation data*

$$\{(\lambda_i, \mathbf{r}_i, \mathbf{w}_i) \mid \lambda_i \in \mathbb{C}, \mathbf{r}_i \in \mathbb{C}^{m\times 1}, \mathbf{w}_i \in \mathbb{C}^{p\times 1},\ i = 1, \cdots, k\}, \tag{4.1}$$

or more compactly

$$\Lambda = \text{diag}\,[\lambda_1, \cdots, \lambda_k] \in \mathbb{C}^{k\times k}, \tag{4.2}$$

$$\mathbf{R} = [\mathbf{r}_1, \cdots, \mathbf{r}_k] \in \mathbb{C}^{m\times k}, \tag{4.3}$$

$$\mathbf{W} = [\mathbf{w}_1, \cdots, \mathbf{w}_k] \in \mathbb{C}^{p\times k}, \tag{4.4}$$

and into the *left interpolation data*

$$\{(\mu_j, \ell_j, \mathbf{v}_j) \mid \mu_j \in \mathbb{C}, \ell_i \in \mathbb{C}^{1\times p}, \mathbf{v}_j \in \mathbb{C}^{1\times m},\ j = 1, \cdots, h, \}, \tag{4.5}$$

or more compactly

$$M = \text{diag}\,[\mu_1, \cdots, \mu_h] \in \mathbb{C}^{h\times h}, \tag{4.6}$$

$$\mathbf{L} = \begin{bmatrix} \ell_1 \\ \vdots \\ \ell_h \end{bmatrix} \in \mathbb{C}^{h\times p}, \tag{4.7}$$

$$\mathbf{V} = \begin{bmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_h \end{bmatrix} \in \mathbb{C}^{h\times m}. \tag{4.8}$$

The rational interpolation problem consists of constructing state-space matrices $[\mathbf{E}, \mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}]$, of appropriate dimensions, such that the associated transfer function $\mathbf{H}(s) = \mathbf{C}(s\mathbf{E} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}$, satisfies both the *right constraints*

$$\mathbf{H}(\lambda_i)\mathbf{r}_i = \left[\mathbf{C}(\lambda_i\mathbf{E} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}\right]\mathbf{r}_i = \mathbf{w}_i,\ i = 1, \cdots, k,$$

and the *left constraints*

$$\ell_j\mathbf{H}(\mu_j) = \ell_j\left[\mathbf{C}(\mu_j\mathbf{E} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}\right] = \mathbf{v}_j,\ j = 1, \cdots, h.$$

The key and novel tools used for studying this problem are the *Loewner matrix*, variously called the *divided-difference matrix* and *null-pole coupling matrix* together with the *shifted Loewner matrix* associated with the data; for the material that follows, we refer to [18] for details on proofs and derivations.

4.1.2 The Loewner and the shifted Loewner matrices

Assume that we are given a set $Z = \{z_1, \cdots, z_{N_s}\}$ of points in the complex plane and the evaluation of a rational matrix function $\mathbf{H}(s)$ at those points: $\{\mathbf{H}(z_1), \cdots, \mathbf{H}(z_{N_s})\}$.

We can partition Z as follows:

$$Z = \{\lambda_1, \cdots, \lambda_k\} \cup \{\mu_1, \cdots, \mu_h\}$$

where $k+h = N_s$. By carefully selecting the direction in which we sample the matrix data, we can construct the right and left interpolation data.

Here the *Loewner matrix* and the *shifted Loewner matrix* are both of dimension $h \times k$, and are defined in terms of the data (4.1) and (4.5) as follows

$$\mathbb{L} = \begin{bmatrix} \frac{\mathbf{v}_1\mathbf{r}_1 - \ell_1\mathbf{w}_1}{\mu_1-\lambda_1} & \cdots & \frac{\mathbf{v}_1\mathbf{r}_k - \ell_1\mathbf{w}_k}{\mu_1-\lambda_k} \\ \vdots & \ddots & \vdots \\ \frac{\mathbf{v}_h\mathbf{r}_1 - \ell_h\mathbf{w}_1}{\mu_h-\lambda_1} & \cdots & \frac{\mathbf{v}_h\mathbf{r}_k - \ell_h\mathbf{w}_k}{\mu_h-\lambda_k} \end{bmatrix}, \tag{4.9}$$

$$\sigma\mathbb{L} = \begin{bmatrix} \frac{\mu_1\mathbf{v}_1\mathbf{r}_1 - \lambda_1\ell_1\mathbf{w}_1}{\mu_1-\lambda_1} & \cdots & \frac{\mu_1\mathbf{v}_1\mathbf{r}_k - \lambda_k\ell_1\mathbf{w}_k}{\mu_1-\lambda_k} \\ \vdots & \ddots & \vdots \\ \frac{\mu_h\mathbf{v}_h\mathbf{r}_1 - \lambda_1\ell_h\mathbf{w}_1}{\mu_h-\lambda_1} & \cdots & \frac{\mu_h\mathbf{v}_h\mathbf{r}_k - \lambda_k\ell_h\mathbf{w}_k}{\mu_h-\lambda_k} \end{bmatrix}. \tag{4.10}$$

Notice that each entry shown above, for instance $\frac{\mathbf{v}_i\mathbf{r}_j - \ell_i\mathbf{w}_j}{\mu_i-\lambda_j}$, is scalar, and is obtained by taking appropriate inner products of the left and right interpolation data. If we assume the existence of a rational matrix function, $\mathbf{H}(s)$, that generates the data then, as the name suggests, the shifted-Loewner matrix is the Loewner matrix corresponding to $s\mathbf{H}(s)$. It can be shown that these matrices satisfy these two Sylvester equations

$$\mathbb{L}\Lambda - M\mathbb{L} = \mathbf{LW} - \mathbf{VR}, \text{ and} \tag{4.11}$$

$$\sigma\mathbb{L}\Lambda - M\sigma\mathbb{L} = \mathbf{LW}\Lambda - M\mathbf{VR}. \tag{4.12}$$

The first consequence of these equations is

Proposition 4.1.1. *There holds:* $\sigma\mathbb{L} - \mathbb{L}\Lambda = \mathbf{VR}$ *and* $\sigma\mathbb{L} - M\mathbb{L} = \mathbf{LW}$.

4.1.3 The solution to the general tangential interpolation problem in the Loewner framework

In this section we will review the conditions for the solution of the general tangential interpolation problem by means of state-space matrices $[\mathbf{E}, \mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}]$, as presented in [18].

Lemma 4.1.1. *Assume that* $k = h$ *and that* $\det(x\mathbb{L} - \sigma\mathbb{L}) \neq 0$, *for all* $x \in \{\lambda_i\} \cup \{\mu_j\}$ *(i.e., the matrix pencil* $(\sigma\mathbb{L}, \mathbb{L})$ *is regular). Then* $\mathbf{E} = -\mathbb{L}$, $\mathbf{A} = -\sigma\mathbb{L}$, $\mathbf{B} = \mathbf{V}$, $\mathbf{C} = \mathbf{W}$ *and* $\mathbf{D} = \mathbf{0}$ *is a minimal realization of an interpolant of the data. That is, the function*

$$\mathbf{H}(s) = \mathbf{W}(\sigma\mathbb{L} - s\mathbb{L})^{-1}\mathbf{V} \tag{4.13}$$

interpolates the data.

Thus, this lemma ensures that the original system is *recovered* after the tangential interpolation data have been constructed and a realization of the system is given in terms of the Loewner matrix pencil together with the $\mathbf{V}$ and $\mathbf{W}$ matrices. Therefore, this method constructs a descriptor-form representation of an underlying dynamical system *exclusively from the data*, just by arranging it in a convenient form. No manipulations of the measurements are involved. For completeness, we quote the proof of this important result.

Proof. Multiplying (4.11) by s and subtracting it from (4.12), we get

$$(\sigma\mathbb{L} - s\mathbb{L})\Lambda - M(\sigma\mathbb{L} - s\mathbb{L}) = \mathbf{LW}(\Lambda - s\mathbf{I}) - (M - s\mathbf{I})\mathbf{VR}. \tag{4.14}$$

Multiplying this equation by $\mathbf{e}_i$ on the right and setting $s = \lambda_i$, we obtain

$$\begin{aligned}(\lambda_i \mathbf{I} - M)(\sigma\mathbb{L} - \lambda_i\mathbb{L})\mathbf{e}_i &= (\lambda_i \mathbf{I} - M)\mathbf{V}\mathbf{r}_i \;\Rightarrow \\ (\sigma\mathbb{L} - \lambda_i\mathbb{L})\mathbf{e}_i &= \mathbf{V}\mathbf{r}_i \;\Rightarrow \\ \mathbf{W}\mathbf{e}_i &= \underbrace{\mathbf{W}(\sigma\mathbb{L} - \lambda_i\mathbb{L})^{-1}\mathbf{V}}_{\mathbf{H}(\lambda_i)}\mathbf{r}_i.\end{aligned}$$

Therefore $\mathbf{w}_i = \mathbf{H}(\lambda_i)\mathbf{r}_i$. This proves the right tangential interpolation property.

To prove the left tangential interpolation property, we multiply the above equation by $\mathbf{e}_j^*$ on the left and set $s = \mu_j$:

$$\begin{aligned}\mathbf{e}_j^*(\sigma\mathbb{L} - \mu_j\mathbb{L})(\mu_j \mathbf{I} - \Lambda) &= \mathbf{e}_j^*\mathbf{L}\mathbf{W}(\mu_j \mathbf{I} - \Lambda) \;\Rightarrow \\ \mathbf{e}_j^*(\sigma\mathbb{L} - \mu_j\mathbb{L}) &= \ell_j\mathbf{W} \;\Rightarrow \\ \mathbf{e}_j^*\mathbf{V} &= \ell_j \underbrace{\mathbf{W}(\sigma\mathbb{L} - \mu_j\mathbb{L})^{-1}\mathbf{V}}_{\mathbf{H}(\mu_j)}\end{aligned}$$

Therefore $\mathbf{v}_j = \ell_j\mathbf{H}(\mu_j)$. ∎

4.2 New contribution

Using the tools described above, one can also address the problem of *recursive interpolation* in the Loewner-matrix framework. Consider the (right and left) interpolation data defined by (4.2)-(4.4) and (4.6)-(4.8) as well as the associated *Loewner matrix* and *shifted Lowener matrix*, as defined in Eqs. (4.9) and (4.10).

4.2.1 The rational interpolation problem

The two matrices satisfy the Sylvester equations

$$\mathbb{L}\Lambda - M\mathbb{L} = \mathbf{LW} - \mathbf{VR} = \begin{bmatrix} \mathbf{L} & \mathbf{V} \end{bmatrix} \begin{bmatrix} \mathbf{W} \\ -\mathbf{R} \end{bmatrix}, \tag{4.15}$$

$$\sigma\mathbb{L}\Lambda - M\sigma\mathbb{L} = \mathbf{LW}\Lambda - M\mathbf{VR} = \begin{bmatrix} \mathbf{L} & M\mathbf{V} \end{bmatrix} \begin{bmatrix} \mathbf{W}\Lambda \\ -\mathbf{R} \end{bmatrix}. \tag{4.16}$$

We now define the $(p+m) \times (p+m)$ rational matrix

$$\Theta(s) = \begin{bmatrix} \mathbf{I}_p & 0 \\ 0 & \mathbf{I}_m \end{bmatrix} + \begin{bmatrix} \mathbf{W} \\ -\mathbf{R} \end{bmatrix} (s\mathbb{L} - \mathbb{L}\Lambda)^{-1} \begin{bmatrix} \mathbf{L} & \mathbf{V} \end{bmatrix} = \begin{bmatrix} \Theta_{11}(s) & \Theta_{12}(s) \\ \Theta_{21}(s) & \Theta_{22}(s) \end{bmatrix}, \tag{4.17}$$

and its inverse

$$\bar{\Theta}(s) = \begin{bmatrix} \mathbf{I}_p & 0 \\ 0 & \mathbf{I}_m \end{bmatrix} + \begin{bmatrix} -\mathbf{W} \\ \mathbf{R} \end{bmatrix} (s\mathbb{L} - M\mathbb{L})^{-1} \begin{bmatrix} \mathbf{L} & \mathbf{V} \end{bmatrix} = \begin{bmatrix} \bar{\Theta}_{11}(s) & \bar{\Theta}_{12}(s) \\ \bar{\Theta}_{21}(s) & \bar{\Theta}_{22}(s) \end{bmatrix}. \tag{4.18}$$

These are the *generating matrices* of the interpolation problem, as all interpolants can be obtained from these quantities (for details see [19], in particular sections 1 and 2). First, we notice that the left and right interpolation conditions are satisfied.

Proposition 4.2.1. *The following relationships hold*

$$\begin{bmatrix} \ell_j & \mathbf{v}_j \end{bmatrix} \Theta(\mu_j) = \mathbf{0}_{1\times(p+m)}, \text{ for } j = 1, \ldots, h, \text{ and} \tag{4.19}$$

$$\bar{\Theta}(\lambda_i) \begin{bmatrix} -\mathbf{w}_i \\ \mathbf{r}_i \end{bmatrix} = \mathbf{0}_{(p+m)\times 1}, \text{ for } i = 1, \ldots, k. \tag{4.20}$$

Proof. We will show the first relationship. The second follows similarly.

Note that

$$\begin{bmatrix} \ell_j & \mathbf{v}_j \end{bmatrix} \boldsymbol{\Theta}(\mu_j) = \begin{bmatrix} \ell_j & \mathbf{v}_j \end{bmatrix} + (\ell_j \mathbf{W} - \mathbf{v}_j \mathbf{R}) (\mu_j \mathbb{L} - \mathbb{L}\Lambda)^{-1} \begin{bmatrix} \mathbf{L} & \mathbf{V} \end{bmatrix}.$$

From the Sylvester equation which the Loewner matrix satisfies (4.11) follows that its j^{th} row is

$$\mathbf{e}_j^*(\mathbf{LW} - \mathbf{VR}) = \ell_j \mathbf{W} - \mathbf{v}_j \mathbf{R} = \mathbf{e}_j^*(\mathbb{L}\Lambda - \mathbf{M}\mathbb{L}) = \mathbf{e}_j^*(\mathbb{L}\Lambda - \mu_j \mathbb{L}) \tag{4.21}$$

$$\Rightarrow (\ell_j \mathbf{W} - \mathbf{v}_j \mathbf{R})(\mathbb{L}\Lambda - \mu_j \mathbb{L})^{-1} = \mathbf{e}_j^*. \tag{4.22}$$

Hence

$$\begin{bmatrix} \ell_j & \mathbf{v}_j \end{bmatrix} \boldsymbol{\Theta}(\mu_j) = \begin{bmatrix} \ell_j & \mathbf{v}_j \end{bmatrix} - \mathbf{e}_j^* \begin{bmatrix} \mathbf{L} & \mathbf{V} \end{bmatrix} = \mathbf{0}_{1 \times (p+m)},$$

which proves the desired equality. ∎

Next, we assert that all interpolants can be obtained as linear matrix fractions constructed from the entries of $\boldsymbol{\Theta}$ and $\bar{\boldsymbol{\Theta}}$.

Theorem 4.2.1. *For any rational matrices* $\mathbf{S}_1(s)$, $\mathbf{S}_2(s)$, $\bar{\mathbf{S}}_1(s)$, $\bar{\mathbf{S}}_2(s)$ *which satisfy the constraint* $\mathbf{S}_1(s)\mathbf{S}_2(s)^{-1} = \bar{\mathbf{S}}_1(s)^{-1}\bar{\mathbf{S}}_2(s)$, *the above result implies that the following rational matrix* $\boldsymbol{\Psi}$ *is an interpolant:*

$$\boldsymbol{\Psi}(s) = \boldsymbol{\Psi}_1(s)\boldsymbol{\Psi}_2(s)^{-1} = [\boldsymbol{\Theta}_{11}(s)\mathbf{S}_1(s) - \boldsymbol{\Theta}_{12}(s)\mathbf{S}_2(s)][-\boldsymbol{\Theta}_{21}(s)\mathbf{S}_1(s) + \boldsymbol{\Theta}_{22}(s)\mathbf{S}_2(s)]^{-1}. \tag{4.23}$$

Similarly, $\boldsymbol{\Psi}$ *can also be written as*

$$\boldsymbol{\Psi}(s) = \bar{\boldsymbol{\Psi}}_1(s)^{-1}\bar{\boldsymbol{\Psi}}_2(s) = [\bar{\mathbf{S}}_1(s)\bar{\boldsymbol{\Theta}}_{11}(s) + \bar{\mathbf{S}}_2(s)\bar{\boldsymbol{\Theta}}_{21}(s)]^{-1}[\bar{\mathbf{S}}_1(s)\bar{\boldsymbol{\Theta}}_{12}(s) + \bar{\mathbf{S}}_2(s)\bar{\boldsymbol{\Theta}}_{22}(s)]. \tag{4.24}$$

The former is a right coprime factorization, while the latter is a left coprime factorization of $\boldsymbol{\Psi}$.

Conversely, given any interpolant which satisfies the left and right tangential interpolation conditions, it can be expressed as Ψ *for* $\mathbf{S}_1(s)$, $\mathbf{S}_2(s)$, $\bar{\mathbf{S}}_1(s)$, $\bar{\mathbf{S}}_2(s)$ *appropriately chosen.*

Proof. We will show the forward direction first. The matrix functions $\mathbf{\Psi}_1(s)$ and $\mathbf{\Psi}_2(s)$ are obtained from

$$\begin{bmatrix} \mathbf{\Theta}_{11}(s) & \mathbf{\Theta}_{12}(s) \\ \mathbf{\Theta}_{21}(s) & \mathbf{\Theta}_{22}(s) \end{bmatrix} \begin{bmatrix} \mathbf{S}_1(s) \\ -\mathbf{S}_2(s) \end{bmatrix} = \begin{bmatrix} \mathbf{\Psi}_1(s) \\ -\mathbf{\Psi}_2(s) \end{bmatrix}. \tag{4.25}$$

Using Eq. (4.19) we have that

$$\begin{bmatrix} \ell_j & \mathbf{v}_j \end{bmatrix} \mathbf{\Theta}(\mu_j) = \mathbf{0} \Rightarrow \begin{bmatrix} \ell_j & \mathbf{v}_j \end{bmatrix} \mathbf{\Theta}(\mu_j) \begin{bmatrix} \mathbf{S}_1 \\ -\mathbf{S}_2 \end{bmatrix} = \mathbf{0} \tag{4.26}$$

$$\Rightarrow \begin{bmatrix} \ell_j & \mathbf{v}_j \end{bmatrix} \begin{bmatrix} \mathbf{\Psi}_1(\mu_j) \\ -\mathbf{\Psi}_2(\mu_j) \end{bmatrix} = \mathbf{0} \tag{4.27}$$

$$\Rightarrow \ell_j \mathbf{\Psi}_1(\mu_j) = \mathbf{v}_j \mathbf{\Psi}_2(\mu_j) \tag{4.28}$$

$$\Rightarrow \ell_j \mathbf{\Psi}_1(\mu_j) \mathbf{\Psi}_2(\mu_j)^{-1} = \mathbf{v}_j, \ \forall j = 1, \ldots, h. \tag{4.29}$$

The above equation shows that the left interpolation conditions are satisfied. Similarly, one can show that the right interpolation conditions are satisfied.

The matrix functions $\bar{\mathbf{\Psi}}_1(s)$ and $\bar{\mathbf{\Psi}}_2(s)$ are obtained from

$$\begin{bmatrix} \bar{\mathbf{S}}_1 & \bar{\mathbf{S}}_2 \end{bmatrix} \begin{bmatrix} \bar{\mathbf{\Theta}}_{11}(s) & \bar{\mathbf{\Theta}}_{12}(s) \\ \bar{\mathbf{\Theta}}_{21}(s) & \bar{\mathbf{\Theta}}_{22}(s) \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{\Psi}}_1(s) & \bar{\mathbf{\Psi}}_2(s) \end{bmatrix}. \tag{4.30}$$

Using Eq. (4.20) we have that

$$\bar{\Theta}(\lambda_i)\begin{bmatrix} -\mathbf{w}_i \\ \mathbf{r}_i \end{bmatrix} = \mathbf{0} \Rightarrow \begin{bmatrix} \bar{\mathbf{S}}_1 & \bar{\mathbf{S}}_2 \end{bmatrix} \bar{\Theta}(\lambda_i)\begin{bmatrix} -\mathbf{w}_i \\ \mathbf{r}_i \end{bmatrix} = \mathbf{0} \tag{4.31}$$

$$\Rightarrow \begin{bmatrix} \bar{\Psi}_1(\lambda_i) & \bar{\Psi}_2(\lambda_i) \end{bmatrix} \begin{bmatrix} -\mathbf{w}_i \\ \mathbf{r}_i \end{bmatrix} = \mathbf{0} \tag{4.32}$$

$$\Rightarrow \bar{\Psi}_2(\lambda_i)\mathbf{r}_i = \bar{\Psi}_1(\lambda_i)\mathbf{w}_i \tag{4.33}$$

$$\Rightarrow \bar{\Psi}_1(\lambda_i)^{-1}\bar{\Psi}_2(\lambda_i)\mathbf{r}_i = \mathbf{w}_i, \ \forall i = 1, \ldots, k. \tag{4.34}$$

Thus we have shown that the right interpolation conditions are satisfied as well.

Next, we have to show that the two expressions for $\Psi(s)$ are the same:

$$\Psi_1(s)\Psi_2(s)^{-1} = \Psi(s) = \bar{\Psi}_1(s)^{-1}\bar{\Psi}_2(s).$$

This is due to the fact that

$$\bar{\Theta}\Theta = \mathbf{I} \Rightarrow \tag{4.35}$$

$$\begin{bmatrix} \bar{\mathbf{S}}_1 & \bar{\mathbf{S}}_2 \end{bmatrix} \bar{\Theta}\Theta \begin{bmatrix} \mathbf{S}_1 \\ -\mathbf{S}_2 \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{S}}_1 & \bar{\mathbf{S}}_2 \end{bmatrix} \begin{bmatrix} \mathbf{S}_1 \\ -\mathbf{S}_2 \end{bmatrix}. \tag{4.36}$$

Recall that $\mathbf{S}_1(s), \mathbf{S}_2(s), \bar{\mathbf{S}}_1(s), \bar{\mathbf{S}}_2(s)$ are chosen such that they satisfy the relationship $\mathbf{S}_1(s)\mathbf{S}_2(s)^{-1} = \bar{\mathbf{S}}_1(s)^{-1}\bar{\mathbf{S}}_2(s)$, so

$$\mathbf{S}_1(s)\mathbf{S}_2(s)^{-1} = \bar{\mathbf{S}}_1(s)^{-1}\bar{\mathbf{S}}_2(s) \Rightarrow \bar{\mathbf{S}}_1(s)\mathbf{S}_1(s) = \bar{\mathbf{S}}_2(s)\mathbf{S}_2(s) \tag{4.37}$$

$$\Rightarrow \begin{bmatrix} \bar{\mathbf{S}}_1 & \bar{\mathbf{S}}_2 \end{bmatrix} \begin{bmatrix} \mathbf{S}_1 \\ -\mathbf{S}_2 \end{bmatrix} = \mathbf{0}. \tag{4.38}$$

Therefore we also have that

$$\underbrace{\begin{bmatrix} \bar{\mathbf{S}}_1 & \bar{\mathbf{S}}_2 \end{bmatrix}\bar{\Theta}}_{\begin{bmatrix} \bar{\Psi}_1 & \bar{\Psi}_2 \end{bmatrix}}\underbrace{\Theta\begin{bmatrix} \mathbf{S}_1 \\ -\mathbf{S}_2 \end{bmatrix}}_{\begin{bmatrix} \Psi_1 \\ -\Psi_2 \end{bmatrix}} = \mathbf{0} \tag{4.39}$$

$$\Rightarrow \bar{\Psi}_1\Psi_1 = \bar{\Psi}_2\Psi_2 \Rightarrow \Psi_1\Psi_2^{-1} = \bar{\Psi}_1^{-1}\bar{\Psi}_2 = \Psi. \tag{4.40}$$

While the first expression for $\Psi(s) = \Psi_1(s)\Psi_2(s)^{-1}$ is more suitable for the left interpolation conditions, the second one, namely $\Psi(s) = \bar{\Psi}_1(s)^{-1}\bar{\Psi}_2(s)$, is more suitable for the right interpolation conditions.

Next we will show the backward direction. Assume that we have constructed the transfer function of an interpolant which satisfies both the left and the right interpolation conditions:

$$\ell_j\mathbf{H}(\mu_j) = \mathbf{v}_j \tag{4.41}$$

$$\mathbf{H}(\lambda_i)\mathbf{r}_i = \mathbf{w}_i. \tag{4.42}$$

Rewriting $\mathbf{H}(s)$ as

$$\mathbf{H}(s) = \mathbf{Q}(s)^{-1}\mathbf{P}(s) = \mathbf{N}(s)\mathbf{D}(s)^{-1}, \tag{4.43}$$

we can express the tangential interpolation conditions as:

$$\ell_j\mathbf{N}(\mu_j)\mathbf{D}(\mu_j)^{-1} = \mathbf{v}_j \Rightarrow \ell_j\mathbf{N}(\mu_j) = \mathbf{v}_j\mathbf{D}(\mu_j) \tag{4.44}$$

$$\mathbf{Q}(\lambda_i)^{-1}\mathbf{P}(\lambda_i)\mathbf{r}_i = \mathbf{w}_i \Rightarrow \mathbf{P}(\lambda_i)\mathbf{r}_i = \mathbf{Q}(\lambda_i)\mathbf{w}_i. \tag{4.45}$$

Define the generating matrix $\Theta(s)$ as in (4.17) and its inverse $\bar{\Theta}(s)$ in (4.18) in the

Loewner-matrix framework. Therefore,

$$\bar{\Theta}\Theta = \mathbf{I}$$

$$\bar{\Theta}\Theta \begin{bmatrix} \mathbf{N} \\ -\mathbf{D} \end{bmatrix} = \begin{bmatrix} \mathbf{N} \\ -\mathbf{D} \end{bmatrix}$$

Thus $\mathbf{H}(s) = \mathbf{N}(s)\mathbf{D}(s)^{-1}$ is an interpolant of the same form as $\mathbf{\Psi}(s) = \mathbf{\Psi}_1(s)\mathbf{\Psi}_2(s)^{-1}$ for $\mathbf{S}_1(s)$ and $\mathbf{S}_2(s)$ obtained from

$$\Theta \begin{bmatrix} \mathbf{S}_1 \\ -\mathbf{S}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{N} \\ -\mathbf{D} \end{bmatrix}$$

$$\bar{\Theta}\Theta \begin{bmatrix} \mathbf{S}_1 \\ -\mathbf{S}_2 \end{bmatrix} = \bar{\Theta} \begin{bmatrix} \mathbf{N} \\ -\mathbf{D} \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{S}_1 \\ -\mathbf{S}_2 \end{bmatrix} = \bar{\Theta} \begin{bmatrix} \mathbf{N} \\ -\mathbf{D} \end{bmatrix}$$

So $\mathbf{S}_1$ and $\mathbf{S}_2$ need to be chosen as: $\mathbf{S}_1 = \bar{\Theta}\mathbf{N}$ and $\mathbf{S}_2 = \bar{\Theta}\mathbf{D}$, respectively. Similarly, we have that

$$\bar{\Theta}\Theta = \mathbf{I} \tag{4.46}$$

$$\begin{bmatrix} \mathbf{Q} & \mathbf{P} \end{bmatrix} \bar{\Theta}\Theta = \begin{bmatrix} \mathbf{Q} & \mathbf{P} \end{bmatrix} \tag{4.47}$$

Thus $\mathbf{H}(s) = \mathbf{Q}(s)^{-1}\mathbf{P}(s)$ is an interpolant of the same form as $\bar{\mathbf{\Psi}}(s) = \bar{\mathbf{\Psi}}_2^{-1}(s)\bar{\mathbf{\Psi}}_1(s)$

for $\bar{\mathbf{S}}_1(s)$ and $\bar{\mathbf{S}}_2(s)$ obtained from

$$\begin{bmatrix} \bar{\mathbf{S}}_1 & \bar{\mathbf{S}}_2 \end{bmatrix} \bar{\mathbf{\Theta}} = \begin{bmatrix} \mathbf{Q} & \mathbf{P} \end{bmatrix}$$
$$\begin{bmatrix} \bar{\mathbf{S}}_1 & \bar{\mathbf{S}}_2 \end{bmatrix} \bar{\mathbf{\Theta}}\mathbf{\Theta} = \begin{bmatrix} \mathbf{Q} & \mathbf{P} \end{bmatrix} \mathbf{\Theta}$$
$$\begin{bmatrix} \bar{\mathbf{S}}_1 & \bar{\mathbf{S}}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{Q} & \mathbf{P} \end{bmatrix} \mathbf{\Theta}$$

So $\bar{\mathbf{S}}_1$ and $\bar{\mathbf{S}}_2$ need to be chosen as: $\bar{\mathbf{S}}_1 = \mathbf{Q}\mathbf{\Theta}$ and $\bar{\mathbf{S}}_2 = \mathbf{P}\mathbf{\Theta}$, respectively.

Clearly, the rational matrices $\mathbf{S}_1(s) = \bar{\mathbf{\Theta}}(s)\mathbf{N}(s)$, $\mathbf{S}_2(s) = \bar{\mathbf{\Theta}}(s)\mathbf{D}(s)$, $\bar{\mathbf{S}}_1(s) = \mathbf{Q}(s)\mathbf{\Theta}(s)$, $\bar{\mathbf{S}}_2(s) = \mathbf{P}(s)\mathbf{\Theta}(s)$ satisfy the constraint $\mathbf{S}_1(s)\mathbf{S}_2(s)^{-1} = \bar{\mathbf{S}}_1(s)^{-1}\bar{\mathbf{S}}_2(s)$ because

$$\mathbf{S}_1(s)\mathbf{S}_2(s)^{-1} = \bar{\mathbf{\Theta}}(s)\mathbf{N}(s)\mathbf{D}(s)^{-1}\mathbf{\Theta}(s) = \bar{\mathbf{\Theta}}(s)\mathbf{H}(s)\mathbf{\Theta}(s) \tag{4.48}$$
$$\bar{\mathbf{S}}_1(s)^{-1}\bar{\mathbf{S}}_2(s) = \bar{\mathbf{\Theta}}(s)\mathbf{Q}(s)^{-1}\mathbf{P}(s)\mathbf{\Theta}(s) = \bar{\mathbf{\Theta}}(s)\mathbf{H}(s)\mathbf{\Theta}(s). \tag{4.49}$$

■

Corollary 4.2.1. *The above proposition implies in particular that for* $\mathbf{S}_1(s) = \mathbf{0}$ *and* $\mathbf{S}_2(s) = -1$, $\mathbf{\Psi} = \mathbf{\Theta}_{12}\mathbf{\Theta}_{22}^{-1}$ *is an interpolant.*

Proof. Here is why:

$$\mathbf{\Theta}_{22} = \mathbf{I} - \mathbf{R}(s\mathbb{L} - \mathbb{L}\mathbf{\Lambda})^{-1}\mathbf{V} \quad \Rightarrow \quad \mathbf{\Theta}_{22}^{-1} = \mathbf{I} + \mathbf{R}(s\mathbb{L} - \mathbb{L}\mathbf{\Lambda} - \mathbf{V}\mathbf{R})^{-1}\mathbf{V}.$$

Hence with $\mathbf{\Theta}_{12} = \mathbf{W}(s\mathbb{L} - \mathbb{L}\mathbf{\Lambda})^{-1}\mathbf{V}$ and making use of the relationship

$$\sigma\mathbb{L} - \mathbb{L}\mathbf{\Lambda} = \mathbf{V}\mathbf{R} \tag{4.50}$$
$$\Rightarrow \sigma\mathbb{L} = \mathbb{L}\mathbf{\Lambda} + \mathbf{V}\mathbf{R} \tag{4.51}$$

it follows that

$$\begin{aligned}
\boldsymbol{\Theta}_{12}\boldsymbol{\Theta}_{22}^{-1} &= \mathbf{W}(s\mathbb{L}-\mathbb{L}\boldsymbol{\Lambda})^{-1}\mathbf{V}[\mathbf{I}+\mathbf{R}(s\mathbb{L}-\mathbb{L}\boldsymbol{\Lambda}-\mathbf{VR})^{-1}\mathbf{V}] && (4.52)\\
&= \mathbf{W}(s\mathbb{L}-\mathbb{L}\boldsymbol{\Lambda})^{-1}\mathbf{V}+\mathbf{W}(s\mathbb{L}-\mathbb{L}\boldsymbol{\Lambda})^{-1}\mathbf{VR}(s\mathbb{L}-\mathbb{L}\boldsymbol{\Lambda}-\mathbf{VR})^{-1}\mathbf{V} && (4.53)\\
&= \mathbf{W}(s\mathbb{L}-\mathbb{L}\boldsymbol{\Lambda})^{-1}\mathbf{V}-\mathbf{W}(s\mathbb{L}-\mathbb{L}\boldsymbol{\Lambda})^{-1}\mathbf{V}+\mathbf{W}(s\mathbb{L}-\mathbb{L}\boldsymbol{\Lambda}-\mathbf{VR})^{-1}\mathbf{V} && (4.54)\\
&= \mathbf{W}(s\mathbb{L}-\mathbb{L}\boldsymbol{\Lambda}-\mathbf{VR})^{-1}\mathbf{V}=\mathbf{W}(s\mathbb{L}-\sigma\mathbb{L})^{-1}\mathbf{V}. && (4.55)
\end{aligned}$$

The latter is the expression of an interpolant given in (4.13). ■

4.2.2 Recursive interpolation in the Loewner framework

If we concatenate two $\boldsymbol{\Theta}$-matrices, which are of the form $\boldsymbol{\Theta}_i = \mathbf{I}+\mathbf{H}_i\boldsymbol{\Phi}_i^{-1}\mathbf{G}_i$, for $i=1,2$, there holds

$$\boldsymbol{\Theta}_1\boldsymbol{\Theta}_2 = \mathbf{I}+\begin{bmatrix}\mathbf{H}_1 & \mathbf{H}_2\end{bmatrix}\begin{bmatrix}\boldsymbol{\Phi}_1 & -\mathbf{G}_1\mathbf{H}_2\\ \mathbf{0} & \boldsymbol{\Phi}_2\end{bmatrix}^{-1}\begin{bmatrix}\mathbf{G}_1\\ \mathbf{G}_2\end{bmatrix}.$$

Now if $\boldsymbol{\Theta}_i$ are defined as in (4.17), there holds

$$\boldsymbol{\Theta}_1(s)\boldsymbol{\Theta}_2(s)=\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}+\begin{bmatrix}\mathbf{W}_1 & \mathbf{W}_2\\ -\mathbf{R}_1 & -\mathbf{R}_2\end{bmatrix}\begin{bmatrix}s\mathbb{L}_1-\mathbb{L}_1\boldsymbol{\Lambda}_1 & -\mathbf{L}_1\mathbf{W}_2+\mathbf{V}_1\mathbf{R}_2\\ \mathbf{0} & s\mathbb{L}_2-\mathbb{L}_2\boldsymbol{\Lambda}_2\end{bmatrix}-1 \tag{4.56}$$

$$\begin{bmatrix}\mathbf{L}_1 & \mathbf{V}_1\\ \mathbf{L}_2 & \mathbf{V}_2\end{bmatrix}=\begin{bmatrix}\boldsymbol{\Theta}_{11}^{12} & \boldsymbol{\Theta}_{12}^{12}\\ \boldsymbol{\Theta}_{21}^{12} & \boldsymbol{\Theta}_{22}^{12}\end{bmatrix}=\boldsymbol{\Theta}^{12}(s). \tag{4.57}$$

Thus

$$\boldsymbol{\Theta}_{12}^{12}=\begin{bmatrix}\mathbf{W}_1 & \mathbf{W}_2\end{bmatrix}\begin{bmatrix}s\mathbb{L}_1-\mathbb{L}_1\boldsymbol{\Lambda}_1 & -\mathbf{L}_1\mathbf{W}_2+\mathbf{V}_1\mathbf{R}_2\\ \mathbf{0} & s\mathbb{L}_2-\mathbb{L}_2\boldsymbol{\Lambda}_2\end{bmatrix}^{-1}\begin{bmatrix}\mathbf{V}_1\\ \mathbf{V}_2\end{bmatrix},$$

while

$$\Theta_{22}^{12} = \mathbf{I} - \begin{bmatrix} \mathbf{R}_1 & \mathbf{R}_2 \end{bmatrix} \begin{bmatrix} s\mathbb{L}_1 - \mathbb{L}_1\Lambda_1 & -\mathbf{L}_1\mathbf{W}_2 + \mathbf{V}_1\mathbf{R}_2 \\ \mathbf{0} & s\mathbb{L}_2 - \mathbb{L}_2\Lambda_2 \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{bmatrix} \Rightarrow$$

$$\left[\Theta_{22}^{12}\right]^{-1} = \mathbf{I} + \begin{bmatrix} \mathbf{R}_1 & \mathbf{R}_2 \end{bmatrix} \begin{bmatrix} s\mathbb{L}_1 - \mathbb{L}_1\Lambda_1 - \mathbf{V}_1\mathbf{R}_1 & -\mathbf{L}_1\mathbf{W}_2 \\ -\mathbf{V}_2\mathbf{R}_1 & s\mathbb{L}_2 - \mathbb{L}_2\Lambda_2 - \mathbf{V}_2\mathbf{R}_2 \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{bmatrix}$$

which implies that the following is an interpolant:

$$\Theta_{22}^{12}\left[\Theta_{22}^{12}\right]^{-1} = \begin{bmatrix} \mathbf{W}_1 & \mathbf{W}_2 \end{bmatrix} \begin{bmatrix} s\mathbb{L}_1 - \mathbb{L}_1\Lambda_1 - \mathbf{V}_1\mathbf{R}_1 & -\mathbf{L}_1\mathbf{W}_2 \\ -\mathbf{V}_2\mathbf{R}_1 & s\mathbb{L}_2 - \mathbb{L}_2\Lambda_2 - \mathbf{V}_2\mathbf{R}_2 \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{bmatrix}. \tag{4.58}$$

Proposition 4.2.2. *The interpolants provided by (4.55) and (4.58) are the same. Hence, there exist nonsingular matrices* $\mathbf{T}_\ell$ *and* $\mathbf{T}_r$*, which are lower and upper (block) triangular, respectively, such that*

$$\begin{aligned} \mathbf{T}_\ell \mathbb{L} \mathbf{T}_r &= \operatorname{diag}\left[\mathbb{L}_1,\ \mathbb{L}_2\right], \\ \mathbf{T}_\ell \sigma\mathbb{L} \mathbf{T}_r &= \begin{bmatrix} \sigma\mathbb{L}_1 & \mathbf{L}_1\mathbf{W}_2 \\ \mathbf{V}_2\mathbf{R}_1 & \sigma\mathbb{L}_2 \end{bmatrix}, \\ \mathbf{W}\mathbf{T}_r &= \begin{bmatrix} \mathbf{W}_1 & \mathbf{W}_2 \end{bmatrix}, \\ \mathbf{T}_\ell \mathbf{V} &= \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{bmatrix}. \end{aligned}$$

In conclusion, given two sets of tangential data points:

$$\left(\Lambda_1 \in \mathbb{C}^{k_1 \times k_1}, \mathbf{R}_1 \in \mathbb{C}^{m \times k_1}, \mathbf{W}_1 \in \mathbb{C}^{p \times k_1}\right) \text{ and } \left(M_1 \in \mathbb{C}^{h_1 \times h_1}, \mathbf{L}_1 \in \mathbb{C}^{h_1 \times p}, \mathbf{V}_1 \in \mathbb{C}^{h_1 \times m}\right),$$

$$\left(\Lambda_2 \in \mathbb{C}^{k_2 \times k_2}, \mathbf{R}_2 \in \mathbb{C}^{m \times k_2}, \mathbf{W}_2 \in \mathbb{C}^{p \times k_2}\right) \text{ and } \left(M_2 \in \mathbb{C}^{h_2 \times h_2}, \mathbf{L}_2 \in \mathbb{C}^{h_2 \times p}, \mathbf{V}_2 \in \mathbb{C}^{h_2 \times m}\right),$$

one can construct the associated Loewner and shifted Loewner matrices $\mathbb{L}_1$, $\sigma\mathbb{L}_1$, for the first set, and $\mathbb{L}_2$, $\sigma\mathbb{L}_2$, for the second set. The recursive interpolation approach in the Loewner matrix framework states that the following matrices

$$\mathbb{L}_r = \operatorname{diag}\left[\mathbb{L}_1, \ \mathbb{L}_2\right], \tag{4.59}$$

$$\sigma\mathbb{L}_r = \begin{bmatrix} \sigma\mathbb{L}_1 & \mathbf{L}_1\mathbf{W}_2 \\ \mathbf{V}_2\mathbf{R}_1 & \sigma\mathbb{L}_2 \end{bmatrix}, \tag{4.60}$$

$$\mathbf{W}_r = \begin{bmatrix} \mathbf{W}_1 & \mathbf{W}_2 \end{bmatrix}, \tag{4.61}$$

$$\mathbf{V}_r = \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{bmatrix} \tag{4.62}$$

correspond to a descriptor-form realization

$$[\mathbf{A} = -\sigma\mathbb{L}_r, \mathbf{E} = -\mathbb{L}_r, \mathbf{B} = \mathbf{V}_r, \mathbf{C} = \mathbf{W}_r, \mathbf{D} = \mathbf{0}]$$

of the system which interpolates both sets of measurements, under the assumption that the first and second data set was interpolated by

$$[\mathbf{A}_1 = -\sigma\mathbb{L}_1, \mathbf{E}_1 = -\mathbb{L}_1, \mathbf{B}_1 = \mathbf{V}_1, \mathbf{C}_1 = \mathbf{W}_1, \mathbf{D}_1 = \mathbf{0}] \text{ and} \tag{4.63}$$

$$[\mathbf{A}_2 = -\sigma\mathbb{L}_2, \mathbf{E}_2 = -\mathbb{L}_2, \mathbf{B}_2 = \mathbf{V}_2, \mathbf{C}_2 = \mathbf{W}_2, \mathbf{D}_2 = \mathbf{0}], \tag{4.64}$$

respectively.

CHAPTER 5

Modeling Multi-Port Scattering Parameters in the Loewner Framework

---

We will present several algorithms based on the theoretical tools developed in the previous chapter.

The data sets that will be analyzed contain N_s samples of the multi-port S-parameters, $\mathbf{S}_i$, at frequency points $j\omega_i$, for $i = 1, \ldots, N_s$. In order to obtain a real system, the condition $\overline{\mathbf{H}(s)} = \mathbf{H}(\bar{s})$ needs to be satisfied. Therefore, the S-parameters at the complex conjugate values of the sample points $-j\omega_i$, $i = 1, \ldots, N_s$, should equal the complex conjugates of the values at $j\omega_i$, namely $\overline{\mathbf{S}_i}$. The devices which are to be modeled are assumed to have the same number of input ports as output ports, denoted by N_p, so for each frequency sample $j\omega_i$, we are supplied with a matrix of dimension $N_p \times N_p$ with complex entries.

We do not assume any other properties of the underlying system, except for stability and bounded-realness.

5.1 Singular value decomposition approach

In practice, the number of given measurements N_s is much larger than the order of the minimal realization of the underlying system, so the condition $\det(x\mathbb{L} - \sigma\mathbb{L}) \neq 0$, for all $x \in \{\lambda_i\} \cup \{\mu_j\}$ in **Lemma** 4.1.1 does not hold. When too many measurements are available, the Loewner pencil $(\sigma\mathbb{L}, \mathbb{L})$, is singular, so $\det(\lambda\mathbb{L} - \sigma\mathbb{L}) = 0$, for all $\lambda \in \mathbb{C}$. In the case of noise-free measurements, the underlying system is obtained by projecting out the singular part, to obtain the regular part of the pencil. The dimension of the regular part is given by the rank of: $\begin{bmatrix} \mathbb{L} & \sigma\mathbb{L} \end{bmatrix}$, $\begin{bmatrix} \mathbb{L} \\ \sigma\mathbb{L} \end{bmatrix}$ or $x\mathbb{L} - \sigma\mathbb{L}$, for any $x \in \{\lambda_i\} \cup \{\mu_j\}$. This observation leads to the first two algorithms presented next.

The first issue to be addressed is the construction of the Loewner and shifted Loewner matrices according to an appropriate partitioning of the data set and an appropriate choice of the sampling directions.

An alternative to the tangential interpolation approach would be the matrix-interpolation approach (for each frequency, we keep the entire S-parameter matrix measurement), but this would lead to Loewner and shifted Loewner matrices of dimention $(N_p \cdot N_s) \times (N_p \cdot N_s)$, which would be expensive to work with, given that typical number of samples in data sets are in the order of 10^3, for number of ports less than 15, and 10^2, for number of ports greater than 15.

5.1.1 Construct the Loewner matrix pencil in the complex approach

Our approach is to select random vectors $\mathbf{r}_i \in \mathbb{C}^{N_p \times 1}$ and set $\ell_i = \mathbf{r}_i^* \in \mathbb{C}^{1 \times N_p}$ (where $(\cdot)^*$ stands for complex conjugate transposed) for each frequency sample $j\omega_i$ and collapse the $N_p \times N_p$ scattering matrix $\mathbf{S}_i$ to either a column vector or a row vector. Random projections are the easiest way to go about this when no underlying structure of the device is known, while choosing the directions in which we sample the S-parameter matrices the same for all frequencies should be avoided.

Remark. The fact that the N_p^2 entries of the matrix are collapsed into one vector of dimension N_p makes this method suitable for devices with a large number of ports.

In summary, the right interpolation data are chosen as

$$\{(\lambda_i = j\omega_i, \mathbf{r}_i, \mathbf{w}_i = \mathbf{S}_i\mathbf{r}_i), \text{ with } \omega_i = 2\pi f_i \in \mathbb{R}, \mathbf{r}_i \in \mathbb{C}^{N_p \times 1}, \mathbf{w}_i \in \mathbb{C}^{N_p \times 1}, \ i = 1, \cdots, N_s\}, \tag{5.1}$$

or more compactly

$$\Lambda = \text{diag}\,[j\omega_1, \cdots, j\omega_{N_s}] \in \mathbb{C}^{N_s \times N_s}, \tag{5.2}$$

$$\mathbf{R} = [\mathbf{r}_1, \cdots, \mathbf{r}_{N_s}] \in \mathbb{C}^{N_p \times N_s}, \tag{5.3}$$

$$\mathbf{W} = [\mathbf{w}_1, \cdots, \mathbf{w}_{N_s}] \in \mathbb{C}^{N_p \times N_s}, \tag{5.4}$$

while the left interpolation data are constructed as

$$\{(\mu_i = -j\omega_i, \ell_i = \mathbf{r}_i^*, \mathbf{v}_i = \ell_i\overline{\mathbf{S}}_i = \mathbf{r}_i^*\overline{\mathbf{S}}_i), \text{ with } \ell_i \in \mathbb{C}^{1 \times N_p}, \mathbf{v}_i \in \mathbb{C}^{1 \times N_p}, \ i = 1, \cdots, N_s\}, \tag{5.5}$$

or more compactly

$$M = \text{diag}\left[-j\omega_1, \cdots, -j\omega_{N_s}\right] \in \mathbb{C}^{N_s \times N_s}, \tag{5.6}$$

$$\mathbf{L} = \begin{bmatrix} \ell_1 \\ \vdots \\ \ell_{N_s} \end{bmatrix} \in \mathbb{C}^{N_s \times N_p}, \tag{5.7}$$

$$\mathbf{V} = \begin{bmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_{N_s} \end{bmatrix} \in \mathbb{C}^{N_s \times N_p}. \tag{5.8}$$

After the tangential data have been constructed, the Loewner and shifted Loewner matrices are built as

$$\mathbb{L} = \begin{bmatrix} \frac{\mathbf{v}_1\mathbf{r}_1 - \ell_1\mathbf{w}_1}{\mu_1 - \lambda_1} & \cdots & \frac{\mathbf{v}_1\mathbf{r}_{N_s} - \ell_1\mathbf{w}_{N_s}}{\mu_1 - \lambda_{N_s}} \\ \vdots & \ddots & \vdots \\ \frac{\mathbf{v}_{N_s}\mathbf{r}_1 - \ell_{N_s}\mathbf{w}_1}{\mu_{N_s} - \lambda_1} & \cdots & \frac{\mathbf{v}_{N_s}\mathbf{r}_{N_s} - \ell_{N_s}\mathbf{w}_{N_s}}{\mu_{N_s} - \lambda_{N_s}} \end{bmatrix}, \tag{5.9}$$

$$\sigma\mathbb{L} = \begin{bmatrix} \frac{\mu_1\mathbf{v}_1\mathbf{r}_1 - \lambda_1\ell_1\mathbf{w}_1}{\mu_1 - \lambda_1} & \cdots & \frac{\mu_1\mathbf{v}_1\mathbf{r}_{N_s} - \lambda_{N_s}\ell_1\mathbf{w}_{N_s}}{\mu_1 - \lambda_{N_s}} \\ \vdots & \ddots & \vdots \\ \frac{\mu_{N_s}\mathbf{v}_{N_s}\mathbf{r}_1 - \lambda_1\ell_{N_s}\mathbf{w}_1}{\mu_{N_s} - \lambda_1} & \cdots & \frac{\mu_{N_s}\mathbf{v}_{N_s}\mathbf{r}_{N_s} - \lambda_{N_s}\ell_{N_s}\mathbf{w}_{N_s}}{\mu_{N_s} - \lambda_{N_s}} \end{bmatrix}. \tag{5.10}$$

The (i,k) entry of the Loewner matrix is

$$\begin{aligned} \mathbb{L}_{i,k} &= \frac{\mathbf{v}_i\mathbf{r}_k - \ell_i\mathbf{w}_k}{\mu_i - \lambda_k} \\ &= \frac{\ell_i\overline{\mathbf{S}}_i\mathbf{r}_k - \ell_i\mathbf{S}_k\mathbf{r}_k}{-\lambda_i - \lambda_k} \\ &= \ell_i\frac{\overline{\mathbf{S}}_i - \mathbf{S}_k}{-\lambda_i - \lambda_k}\mathbf{r}_k \end{aligned}$$

while that of the shifted Loewner matrix is

$$\begin{aligned}\sigma\mathbb{L}_{i,k} &= \frac{\mu_i \mathbf{v}_i \mathbf{r}_k - \lambda_k \ell_i \mathbf{w}_k}{\mu_i - \lambda_k} \\ &= \frac{-\lambda_i \ell_i \overline{\mathbf{S}}_i \mathbf{r}_k - \lambda_k \ell_i \mathbf{S}_k \mathbf{r}_k}{-\lambda_i - \lambda_k} \\ &= \ell_i \frac{-\lambda_i \overline{\mathbf{S}}_i - \lambda_k \mathbf{S}_k}{-\lambda_i - \lambda_k} \mathbf{r}_k.\end{aligned}$$

Instead of choosing random sampling directions $\mathbf{r}_i$ and setting $\ell_i = \mathbf{r}_i^*$, one may invest some computational time in choosing them in a clever way, avoiding the situation in which the same degree-interpolant is different every time the algorithm is run. The price to be paid for this is introducing a factor of N_p^3 in the computational complexity of the algorithm, compared to N_p^2 for random projections. A good way of choosing the directions is as the transpose of the sum of the left singular vectors and as the complex conjugated sum of the right singular vectors of the matrix measurement at the sample point λ_i, so computing the singular value decomposition of each matrix measurement introduces the N_p^3 scaling factor in the complexity:

$$\mathbf{S}_i = \mathbf{X}\Sigma\mathbf{Y}^* = \sum_{k=1}^{N_p} \sigma_k \mathbf{x}_k \mathbf{y}_k^*.$$

Therefore,

$$\mathbf{S}_i \underbrace{\sum_{k=1}^{N_p} \mathbf{y}_k}_{\mathbf{r}_i} = \underbrace{\sum_{k=1}^{N_p} \sigma_k \mathbf{x}_k}_{\mathbf{w}_i}. \tag{5.11}$$

We also have that

$$\bar{\mathbf{S}}_i = \bar{\mathbf{X}}\Sigma\bar{\mathbf{Y}}^* = \sum_{k=1}^{N_p} \sigma_k \bar{\mathbf{x}}_k \bar{\mathbf{y}}_k^*.$$

Therefore,

$$\underbrace{\sum_{k=1}^{N_p} \bar{\mathbf{x}}_k^*}_{\ell_i} \bar{\mathbf{S}}_i = \underbrace{\sum_{k=1}^{N_p} \sigma_k \bar{\mathbf{y}}_k^*}_{\mathbf{v}_i}. \tag{5.12}$$

Hence, the right sampling directions are taken as the sum of the complex conjugated right singular vectors, while the left sampling directions are given by the sum of the transposed left singular vectors. In this case, the left and right projectors do not satisfy the relationship $\ell_i = \mathbf{r}_i^*$ any more.

5.1.2 Singular value decomposition of the Loewner matrix pencil in the complex approach

As **Algorithm 1**, we present the main result of [18], namely the construction of state space models of the form $[\mathbf{E}, \mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{0}]$ (with $\mathbf{D} = 0$) directly from input/output measurements. For $\mathbf{D} \neq 0$, see [18]. We will present an example in chapter 7 for which we can recover the original system which has a non-zero $\mathbf{D}$ matrix by contructing a model of the size N_p more the dimension of the original state-space matrices $\mathbf{A}$ and $\mathbf{E}$.

If the conditions of Lemma 4.1.1 are not satisfied, we proceed by appropriately projecting $\mathbb{L}$ and $\sigma\mathbb{L}$. The following is the main assumption pertaining to the construction proposed:

$$\operatorname{rank}(x\mathbb{L} - \sigma\mathbb{L}) = \operatorname{rank}\begin{bmatrix} \mathbb{L} & \sigma\mathbb{L} \end{bmatrix} = \operatorname{rank}\begin{bmatrix} \mathbb{L} \\ \sigma\mathbb{L} \end{bmatrix} =: k, \quad x \in \{\lambda_i\} \cup \{\mu_j\} \tag{5.13}$$

If these conditions are satisfied, we compute the *singular value decomposition*

$$x\mathbb{L} - \sigma\mathbb{L} = \mathbf{Y}\Sigma\mathbf{X}, \tag{5.14}$$

where $\operatorname{rank}(x\mathbb{L} - \sigma\mathbb{L}) = \operatorname{rank}(\Sigma) =: k$, $\mathbf{Y} \in \mathbb{C}^{N_s \times k}$ and $\mathbf{X} \in \mathbb{C}^{k \times N_s}$, where k is the *dimension of the regular part* of $x\mathbb{L} - \sigma\mathbb{L}$ and it depends on the application at hand. Alternatively, one can choose k, the dimension of the reduced system based on the desired accuracy. The accuracy of the sub-optimal systems can be estimated from the

drop of the singular values of the pencil $\sigma\mathbb{L} - \lambda\mathbb{L}$, where λ is chosen as any frequency sample.

Theorem 5.1.1. *With the quantities above, if the order of the reduced system is chosen as the dimension of the regular part, a minimal realization* $[\mathbf{E}, \mathbf{A}, \mathbf{B}, \mathbf{C}]$*, of an interpolant is given as follows:*

$$\left.\begin{aligned} \mathbf{E} &:= -\mathbf{Y}^*\mathbb{L}\mathbf{X}^* \\ \mathbf{A} &:= -\mathbf{Y}^*\sigma\mathbb{L}\mathbf{X}^* \\ \mathbf{B} &:= \mathbf{Y}^*\mathbf{V} \\ \mathbf{C} &:= \mathbf{W}\mathbf{X}^* \end{aligned}\right\} \tag{5.15}$$

Assumption (5.13) demands that the system has full rank tangential generalized controllability and observability matrices, and is generically always satisfied.

5.1.3 Construct the Loewner matrix pencil in the real approach

Even though the previous approach of constructing the matrices uses the information at the complex conjugate values of the sample points, the resulting matrices are complex and their eigenvalues will not come in complex conjugate pairs due to the fact that the information at the complex conjugate frequency samples is used as left interpolation data, while the information at the original frequency samples is used as right interpolation data [20].

In order to guarantee a resulting system which is real, the right interpolation data is chosen as

$$\left\{ \left(j\omega_i, -j\omega_i, \mathbf{r}_i, \bar{\mathbf{r}}_i, \mathbf{w}_i = \mathbf{S}_i\mathbf{r}_i, \bar{\mathbf{w}}_i = \bar{\mathbf{S}}_i\bar{\mathbf{r}}_i\right), \right.$$

$$\left. \text{with } \omega_i = 2\pi f_i \in \mathbb{R}, \mathbf{r}_i \in \mathbb{C}^{N_p \times 1}, \mathbf{w}_i \in \mathbb{C}^{N_p \times 1},\ i = 1, \cdots, \frac{N_s}{2} \right\},$$

or more compactly

$$\Lambda = \text{diag}\left[j\omega_1,\ -j\omega_1,\ \cdots,\ j\omega_{\frac{N_s}{2}},\ -j\omega_{\frac{N_s}{2}}\right] \in \mathbb{C}^{N_s \times N_s}, \tag{5.16}$$

$$\mathbf{R} = \left[\mathbf{r}_1,\ \bar{\mathbf{r}}_1,\ \cdots,\ \mathbf{r}_{\frac{N_s}{2}},\ \bar{\mathbf{r}}_{\frac{N_s}{2}}\right] \in \mathbb{C}^{N_p \times N_s}, \tag{5.17}$$

$$\mathbf{W} = \left[\mathbf{w}_1,\ \bar{\mathbf{w}}_1,\ \cdots,\ \mathbf{w}_{\frac{N_s}{2}},\ \bar{\mathbf{w}}_{\frac{N_s}{2}}\right] \in \mathbb{C}^{N_p \times N_s}, \tag{5.18}$$

while the left interpolation data is constructed as

$$\left\{\left(j\omega_{i+\frac{N_s}{2}},\ -j\omega_{i+\frac{N_s}{2}},\ \ell_i,\ \bar{\ell}_i,\ \mathbf{v}_i = \ell_i \mathbf{S}_{i+\frac{N_s}{2}},\ \bar{\mathbf{v}}_i = \bar{\ell}_i \overline{\mathbf{S}}_{i+\frac{N_s}{2}}\right),\right.$$

$$\left.\text{with } \ell_i \in \mathbb{C}^{1\times N_p}, \mathbf{v}_i \in \mathbb{C}^{1\times N_p},\ i = 1, \cdots, \frac{N_s}{2}\right\},$$

or more compactly

$$M = \text{diag}\left[j\omega_{1+\frac{N_s}{2}},\ -j\omega_{1+\frac{N_s}{2}},\ \cdots,\ j\omega_{N_s},\ -j\omega_{N_s}\right] \in \mathbb{C}^{N_s \times N_s}, \tag{5.19}$$

$$\mathbf{L} = \begin{bmatrix} \ell_1 \\ \bar{\ell}_1 \\ \vdots \\ \ell_{\frac{N_s}{2}} \\ \bar{\ell}_{\frac{N_s}{2}} \end{bmatrix} \in \mathbb{C}^{N_s \times N_p}, \tag{5.20}$$

$$\mathbf{V} = \begin{bmatrix} \mathbf{v}_1 \\ \bar{\mathbf{v}}_1 \\ \vdots \\ \mathbf{v}_{\frac{N_s}{2}} \\ \bar{\mathbf{v}}_{\frac{N_s}{2}} \end{bmatrix} \in \mathbb{C}^{N_s \times N_p}. \tag{5.21}$$

Without loss of generality, we assumed that we have an even number of samples. The vectors ℓ_i and $\mathbf{r}_i$ are chosen as random complex vectors. After the tangential data

have been constructed, the Loewner and shifted Loewner matrices are built according to formulas Eq. (5.9)-(5.10).

Instead of choossing the sampling directions ℓ_i and $\mathbf{r}_i$ random, one may invest some computational time in choosing them in a clever way, avoiding the situation in which the same degree-interpolant is different every time the algorithm is run. A good way of choosing the directions is as sum of the left and right singular vectors of the matrix measurement at the sample point λ_i, together with their complex conjugate:

$$\mathbf{S}_i = \mathbf{X}\Sigma\mathbf{Y}^* = \sum_{k=1}^{N_p} \sigma_k \mathbf{x}_k \mathbf{y}_k^*, \text{ for } i = 1, \ldots, \frac{N_s}{2}.$$

Therefore,

$$\mathbf{S}_i \underbrace{\sum_{k=1}^{N_p} \mathbf{y}_k}_{\mathbf{r}_i} = \underbrace{\sum_{k=1}^{N_p} \sigma_k \mathbf{x}_k}_{\mathbf{w}_i}. \tag{5.22}$$

We also need to add the complex conjugate of those values, as

$$\left\{ \left(j\omega_i, -j\omega_i, \mathbf{r}_i = \sum_{k=1}^{N_p} \mathbf{y}_k, \bar{\mathbf{r}}_i = \sum_{k=1}^{N_p} \bar{\mathbf{y}}_k, \mathbf{w}_i = \sum_{k=1}^{N_p} \sigma_k \mathbf{x}_k, \bar{\mathbf{w}}_i = \sum_{k=1}^{N_p} \sigma_k \bar{\mathbf{x}}_k \right), \right.$$

$$\left. \text{with } \omega_i = 2\pi f_i \in \mathbb{R}, \mathbf{r}_i \in \mathbb{C}^{N_p \times 1}, \mathbf{w}_i \in \mathbb{C}^{N_p \times 1}, \ i = 1, \cdots, \frac{N_s}{2} \right\}.$$

For the other half of the measurements we have

$$\mathbf{S}_i = \mathbf{X}\Sigma\mathbf{Y}^* = \sum_{k=1}^{N_p} \sigma_k \mathbf{x}_k \mathbf{y}_k^*, \text{ for } i = \frac{N_s}{2} + 1, \ldots, N_s.$$

Therefore,

$$\underbrace{\sum_{k=1}^{N_p} \mathbf{x}_k^* \mathbf{S}_i}_{\ell_{i-\frac{N_s}{2}}} = \underbrace{\sum_{k=1}^{N_p} \sigma_k \mathbf{y}_k^*}_{\mathbf{v}_{i-\frac{N_s}{2}}}. \tag{5.23}$$

We also need to add the complex conjugate of those values, as

$$\left\{\left(j\omega_i, -j\omega_i, \ell_{i-\frac{N_s}{2}} = \sum_{k=1}^{N_p} \mathbf{x}_k^*, \bar{\ell}_{i-\frac{N_s}{2}} = \sum_{k=1}^{N_p} \bar{\mathbf{x}}_k^*, \mathbf{v}_{i-\frac{N_s}{2}} = \sum_{k=1}^{N_p} \sigma_k \mathbf{y}_k^*, \bar{\mathbf{v}}_{i-\frac{N_s}{2}} = \sum_{k=1}^{N_p} \sigma_k \bar{\mathbf{y}}_k^*\right),\right.$$

$$\left.\text{with } \omega_i = 2\pi f_i \in \mathbb{R}, \ell_{i-\frac{N_s}{2}} \in \mathbb{C}^{1\times N_p}, \mathbf{v}_{i-\frac{N_s}{2}} \in \mathbb{C}^{1\times N_p},\ i = \frac{N_s}{2}+1, \ldots, N_s\right\}.$$

5.1.4 Singular value decomposition of the Loewner matrix pencil in the real approach

If the conditions of **Lemma** 4.1.1 are not satisfied (the Loewner matrix pencil is singular due to the fact that the number of measurements N_s is larger than needed), we proceed by appropriately projecting $\mathbb{L}$ and $\sigma\mathbb{L}$. Eq. (5.13) is the main assumption pertaining to the construction proposed. If these conditions are satisfied, we compute the *singular value decomposition*

$$x\mathbb{L} - \sigma\mathbb{L} = \mathbf{Y}\Sigma\mathbf{X}, \tag{5.24}$$

where $\text{rank}\,(x\mathbb{L} - \sigma\mathbb{L}) = \text{rank}\,(\Sigma) =: k$, $\mathbf{Y} \in \mathbb{C}^{N_s\times k}$ and $\mathbf{X} \in \mathbb{C}^{k\times N_s}$, where k is the *dimension of the regular part* of $x\mathbb{L} - \sigma\mathbb{L}$ and depends on the application at hand, or can be chosen as the desired dimension of the macromodel.

Theorem 5.1.2. *With the quantities above, if the order of the reduced system is chosen as the dimension of the regular part, a minimal realization* $[\mathbf{E}, \mathbf{A}, \mathbf{B}, \mathbf{C}]$*, of an interpolant is given as follows:*

$$\left.\begin{aligned} \mathbf{E} &:= -\mathbf{Y}^*\mathbb{L}\mathbf{X}^* \\ \mathbf{A} &:= -\mathbf{Y}^*\sigma\mathbb{L}\mathbf{X}^* \\ \mathbf{B} &:= \mathbf{Y}^*\mathbf{V} \\ \mathbf{C} &:= \mathbf{W}\mathbf{X}^* \end{aligned}\right\} \tag{5.25}$$

Assumption (5.13) demands that the system has full rank tangential generalized controllability and observability matrices, and is generically always satisfied.

The procedure described above constitutes the **second algorithm** we are proposing.

5.1.5 Bound on singular value decay of solutions to Sylvester equations

The theory developped in [21] on bounds on the decay rate of the singular values of solutions to Sylvester equations was one of the tools we used in analyzing the numerical examples from the point of view of algorithms 1 and 2. For completeness, we present **Theorem 2.1.1.** from the original thesis:

Theorem 1. *Let* $\mathbf{A} \in \mathbb{C}^{n\times n}$, $\mathbf{B} \in \mathbb{C}^{m\times m}$ *be stable and Hermitian. Let* $[a, b]$ *and* $[c, d]$ *be intervals containing the eigenvalues of* $\mathbf{A}$ *and* $\mathbf{B}$*, respectively. Define*

$$\eta = 2\frac{(b-a)(d-c)}{(a+c)(b+d)}. \tag{5.26}$$

Additionally, let $\mathbf{C} \in \mathbb{C}^{n\times p}$ *and* $\mathbf{D} \in \mathbb{C}^{p\times m}$*, where* $p < \min(m, n)$*. The singular values* $\sigma_i(\mathbf{X})$ *of the solution to the Sylvester equation* $\mathbf{AX}+\mathbf{XB} = -\mathbf{CD}^*$ *obey the following bound for* $1 \leq pr < n$*:*

$$\frac{\sigma_{pr+1}(\mathbf{X})}{\sigma_1(\mathbf{X})} \leq \left(\frac{1-\sqrt{k'_r}}{1+\sqrt{k'_r}}\right)^2. \tag{5.27}$$

In the above, k'_r *relates to the nome* q^r *by way of*

$$\sqrt{k'} = \frac{1-2q+2q^4-2q^9+\ldots}{1+2q+2q^4+2q^9+\ldots} \tag{5.28}$$

where q is the nome corresponding to the complementary elliptic modulus

$$k' = \frac{1}{1+\eta+\sqrt{\eta(\eta+2)}}.$$

If $\mathbf{B} = \mathbf{A} = \mathbf{A}^*$, we can approximate the bound using Eq. (2.13) in the original thesis

$$\frac{\sigma_{pr+1}(\mathbf{X})}{\sigma_1(\mathbf{X})} \lesssim 4e^{-2\pi r\frac{K'}{K}}, \tag{5.29}$$

where $K = u(1)$ is the *complete elliptic integral*, K' is the *complementary complete elliptic integral*, the complementary modulus $k' = \frac{1}{\kappa(\mathbf{A})}$ and $\kappa(\mathbf{A}) = \|\mathbf{A}\|_2 \|\mathbf{A}^{-1}\|_2$.

Our goal is to apply the bound (5.29) in order to predict the decay of the singular values of the Loewner and shifted Loewner matrices and, consequently, an appropriate choice for the order of the reduced system, without having to go through the costly process of computing the singular values themselves.

Recall that the Loewner and shifted Loewner matrices satisfy the two Sylvester equations:

$$\mathbb{L}\Lambda - M\mathbb{L} = \mathbf{LW} - \mathbf{VR}, \text{ and} \tag{5.30}$$

$$\sigma\mathbb{L}\Lambda - M\sigma\mathbb{L} = \mathbf{LW}\Lambda - M\mathbf{VR}. \tag{5.31}$$

Moreover, recall the complex approach of constructing the matrices with the matrices Λ and M taken as:

$$\Lambda = \text{diag}\,[j\omega_1, \cdots, j\omega_{N_s}] \in \mathbb{C}^{N_s \times N_s}, \tag{5.32}$$

$$M = \text{diag}\,[-j\omega_1, \cdots, -j\omega_{N_s}] \in \mathbb{C}^{N_s \times N_s}. \tag{5.33}$$

In order for the condition that Λ and M be stable to be satisfied, we have to multiply

Eq. (5.30) and (5.31) by $-\frac{1}{j}$ to obtain

$$\mathbf{A} = M/j = \text{diag}\,[-\omega_1,\ \cdots,\ -\omega_{N_s}] \in \mathbb{R}^{N_s \times N_s}, \tag{5.34}$$

$$\mathbf{B} = -\Lambda/j = \text{diag}\,[-\omega_1,\ \cdots,\ -\omega_{N_s}] \in \mathbb{R}^{N_s \times N_s} \text{ and} \tag{5.35}$$

$$-\mathbf{CD}^* = -\frac{\mathbf{LW} - \mathbf{VR}}{j} \text{ for (5.30) and} \tag{5.36}$$

$$-\mathbf{CD}^* = -\frac{\mathbf{LW}\Lambda - M\mathbf{VR}}{j} \text{ for (5.31).} \tag{5.37}$$

Therefore, our matrices satisfy the condition $\mathbf{B} = \mathbf{A} = \mathbf{A}^*$, so the approximated bound in equation (5.29) can be applied with $\kappa(\mathbf{A}) = \frac{\omega_{N_s}}{\omega_1}$, assuming that the measurements are given for increasing values of the frequencies.

Recall that in the real approach, the measurements are divided in two: the first half, together with their complex conjugate pairs are used as right interpolation data, while the other half, together with their complex conjugate pairs, are used as left interpolation data, so the Λ and M matrices are taken as:

$$\Lambda = \text{diag}\left[j\omega_1,\ -j\omega_1, \cdots,\ j\omega_{\frac{N_s}{2}},\ -j\omega_{\frac{N_s}{2}}\right] \in \mathbb{C}^{N_s \times N_s}, \tag{5.38}$$

$$M = \text{diag}\left[j\omega_{\frac{N_s}{2}+1},\ -j\omega_{\frac{N_s}{2}+1},\ \cdots,\ j\omega_{N_s},\ -j\omega_{N_s}\right] \in \mathbb{C}^{N_s \times N_s} \tag{5.39}$$

This clearly shows that Λ and M cannot be made stable by scaling due to the fact that the diagonal entries have alternating signs, so the bound cannot be applied to predict the decay rate of the singular values when the real construction of the Loewner pencil is used.

5.1.6 The Kronecker canonical form of a singular matrix pencil

As stated previously, the Loewner matrix pencil is singular when too many measurements are available. Therefore, we can compute the so-called *Kronecker canonical*

form of the pencil to separate the finite part of the pencil from the singular part. This can be performed using the GUPTRI software [22], [23] which computes the generalized upper triangular form $\mathbf{S} - \lambda\mathbf{T}$ of a matrix pencil $\mathbf{A} - \lambda\mathbf{B}$, via unitary equivalence transformations $\mathbf{P}$ and $\mathbf{Q}$: $\mathbf{S} - \lambda\mathbf{T} = \mathbf{P}^*(\mathbf{A} - \lambda\mathbf{B})\mathbf{Q}$. The resulting $\mathbf{S}$ and $\mathbf{T}$ matrices will be of the form

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}_r & * & * & * & * \\ \mathbf{0} & \mathbf{S}_z & * & * & * \\ \mathbf{0} & \mathbf{0} & \mathbf{S}_f & * & * \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{S}_i & * \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{S}_l \end{bmatrix}, \mathbf{T} = \begin{bmatrix} \mathbf{T}_r & * & * & * & * \\ \mathbf{0} & \mathbf{T}_z & * & * & * \\ \mathbf{0} & \mathbf{0} & \mathbf{T}_f & * & * \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{T}_i & * \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{T}_l \end{bmatrix}, \tag{5.40}$$

where the diagonal blocks describe the Kronecker canonical form of the pencil $\mathbf{A} - \lambda\mathbf{B}$ as follows:

1. $\mathbf{S}_r - \lambda\mathbf{T}_r$ contains the right singular structure
2. $\mathbf{S}_z - \lambda\mathbf{T}_z$ contains the Jordan structure for the 0 eigenvalue
3. $\mathbf{S}_f - \lambda\mathbf{T}_f$ contains the Jordan structure for the finite nonzero eigenvalues
4. $\mathbf{S}_i - \lambda\mathbf{T}_i$ contains the Jordan structure for the infinite eigenvalue
5. $\mathbf{S}_l - \lambda\mathbf{T}_l$ contains the left singular structure.

The software has a Matlab implementation available with the following function call

```
[S, T, P, Q, kstr] = guptri(A, B, EPSU,  GAP,  ZERO)
```

The input variables are: `EPSU` as the relative uncertainty in the data (by default `EPSU = 1E-8`), `GAP`, which is used in singular value decompositions to make rank decisions by searching for adjacent singular values whose ratios exceed `GAP` (`GAP` should be between 1 and 1000; by default `GAP = 1000`) and last, `ZERO`, which is used as a

logical variable which when it is true (nonzero) forces GUPTRI to zero out small singular values during the reduction process; ZERO= 0 (false) will give a reduced pencil made by true equivalence transformations, but the pencil will not exactly have the structure described by `kstr` (by default, ZERO = 1). For more details, see [22] and [23].

The complete Kronecker structure of the pencil is revealed by `kstr` as:

- the first part of `kstr` (the columns to the left of the first -1) shall be interpreted as follows: the number of right singular blocks of size i is $\texttt{kstr}(1, i+1) - \texttt{kstr}(2, i+1)$ and the number of Jordan blocks of dimension i corresponding to the zero eigenvalue is $\texttt{kstr}(2, i) - \texttt{kstr}(1, i+1)$
- the second part of `kstr` contains the corresponding information about the left singular part and the blocks corresponding to the infinite eigenvalues
- the third part of `kstr` contains information about the size of the finite non-zero part.

5.2 Adaptive approach

In the upcoming section we will use another approach which allows us to treat the problem of the singular matrix pencil obtained after building the Loewner and shifted Loewner matrices in a suitable way, using all the N_s available measurements. The adaptive approach is an alternative to computing the singular value decomposition or the Kronecker canonical form of the singular Loewner matrix pencil $(\sigma\mathbb{L}, \mathbb{L})$ constructed from all the measurements. The proposed algorithms described in this section choose only a certain number of samples from the available ones in an *adaptive* fashion and use them to construct the desired model. Therefore, we start with a low order model, compute the errors and add new measurements where the largest errors

occur. We use the new measurements to update our model and, in case the errors are still high, we repeat the procedure: compute the errors, select new measurements and update the model.

The most common technique would be to assess the error between the model and the data in the magnitude of all the entries of the $N_p \times N_p$ scattering matrix for each frequency sample. Keeping in mind that the goal is to apply these algorithms to devices with a large number of ports N_p, checking the magnitude of *all* entries, for *all* samples, is computationally expensive. We choose to use other quantities instead, namely the singular values of the error matrices, computed as the difference between the transfer function of the current model evaluated at that particular frequency sample and the corresponding S-parameter data matrix. The N_p singular values of the error matrix contain information about the errors in all entries and, moreover, have the advantage that they are real and positive. Recall that the $\mathcal{H}_\infty$ norm of a matrix is given by the largest singular value, so using the singular values of the error matrices as an error measure is justified. If these singular values are small, the errors will be small as well and the model will be of good quality. Otherwise, we have a poor model.

5.2.1 Complex adaptive approach

The procedure starts by computing a system of order N_p from N_p measurements selected from the N_s available. The N_p indices for these measurements are linearly distributed between 1 and N_s. The system is constructed by building the Λ, M, $\mathbf{R}$, $\mathbf{L}$, $\mathbf{W}$, $\mathbf{V}$, $\mathbb{L}$ and $\sigma\mathbb{L}$ matrices as described in section 5.1.1 (with the sampling directions ℓ_i and $\mathbf{r}_i$ chosen as outlined in Eq. (5.11) and (5.12)) and setting $\mathbf{E}$ as $-\mathbb{L}$, $\mathbf{A}$ as $-\sigma\mathbb{L}$, $\mathbf{B}$ as $\mathbf{V}$, $\mathbf{C}$ as $\mathbf{W}$ and $\mathbf{D}$ as $\mathbf{0}$. Starting with a system of dimension N_p ensures that all the singular values of the intermediary model are non-zero.

At each sample frequency point, the N_p singular values of the error matrix obtained

as the difference between the S-parameter data matrix and the transfer function of the current model evaluated at that particular sample are computed. The next measurement is chosen at the frequency where the maximum error (singular value) occurs. At this stage we keep a record of the measurements which have already been selected and use new samples in the future.

The sampling directions for the new measurement can be chosen as

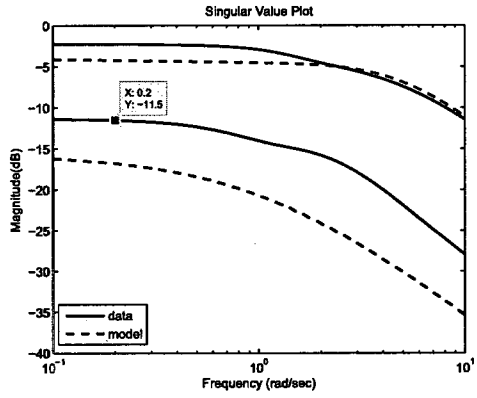
- random complex vectors (**algorithm 3**)
- the singular vectors associated with the largest singular value of the error matrix (**algorithm 5**)
- the sum of the singular vectors of the error matrix which gives the largest error (**algorithm 7**)

At the next step, a system of order $N_p + 1$ is constructed from the initial N_p measurements together with the one which leads to the largest error by following the steps described in section 5.1.1. We preserve the system built up to this stage and only append the last row or column obtained from combining the new measurement with all the previous ones. The singular values of the error matrix are computed again and a new measurement is chosen according to the same criterion as before (has not already been used and gives the largest error). The procedure continues until the desired accuracy or the desired order of the reduced model has been reached.

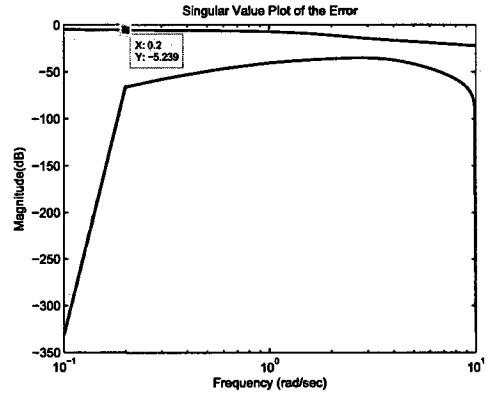
Let us make the steps of these algorithms more clear by considering the following example. The main ideas are common, but we will explain algorithm 5 in more detail. A multiple-input multiple-output (MIMO) bounded-real system is given by the matrix transfer function

$$\mathbf{H}(s) = \begin{bmatrix} \frac{1}{2.5(s+1)} & \frac{1}{1.3[(s+1)^2+1]} \\ \frac{1}{[(s+2)^2+3^2]} & \frac{3}{s+5} \end{bmatrix}.$$

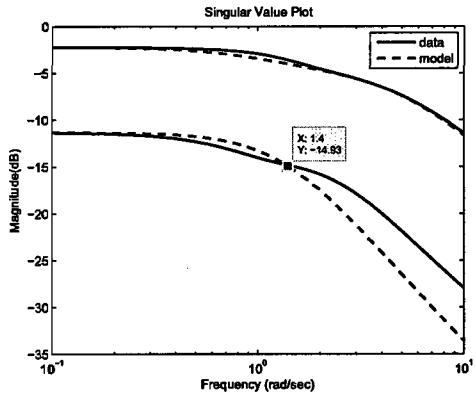
The system is of order 6 with poles at $-1 \pm j$, $-2 \pm 3j$, -1 and -5. We sample the transfer function at 100 linearly spaced frequency points between 10^{-1} rad/sec and 10^{1} rad/sec and assume noise-free measurements.



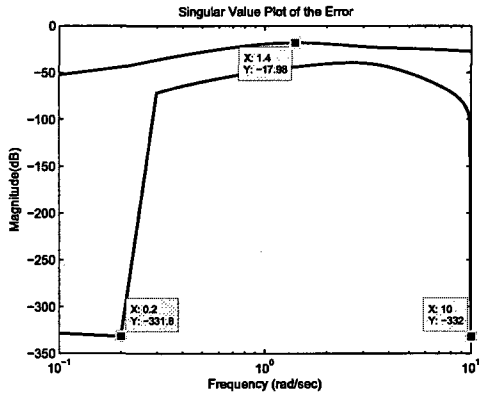
(a) Order 2



(b) Order 2



(c) Order 3

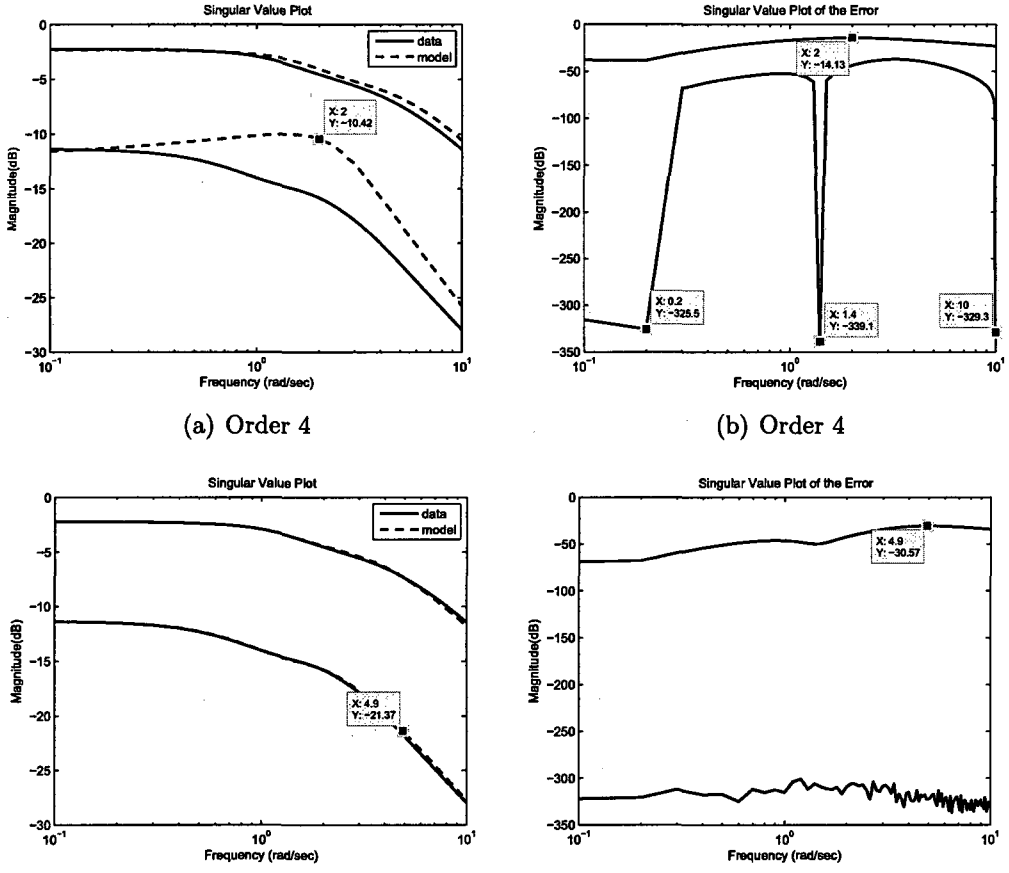


(d) Order 3

Figure 5.1: Intermediary models of order 2 and 3

We start with a system of order N_p, which is 2 in this case, obtained using the first and last measurement with the tangential directions constructed from the SVD of the scattering matrices. We plot it against the data in Figure 5.1(a). Figure 5.1(b) shows the two singular values of all the error matrices. We choose the next measurement at the frequency where the largest singular value in the error matrices occurs (0.2 in this case), with the sampling directions of the new measurement taken as the singular vectors associated with that particular singular value of the error matrix. Thus we

obtain a third order system shown in Figure 5.1(c). The error plot in Figure 5.1(d) shows that the second singular value of the error matrices is very small at all the frequencies which we have already used (0.1, 0.2 and 10 rad/sec).



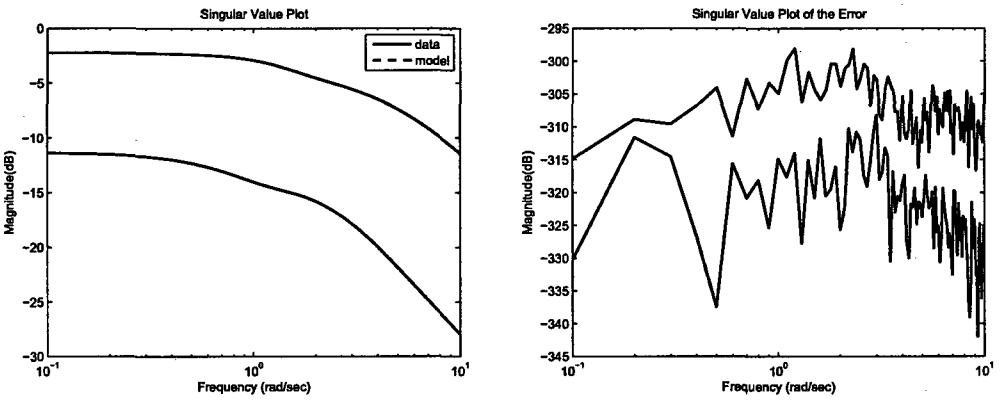
(a) Order 4 (b) Order 4

(c) Order 5 (d) Order 5

Figure 5.2: Intermediary models of order 4 and 5

We choose the next measurement at 1.4 rad/sec, where the largest error in the singular values occurs and obtain a fourth order system shown in Figure 5.2(a). The singular values of the error matrices suggest to choose the next measurement at 2 rad/sec with the sampling directions taken as the singular vectors associated with the singular value 2 of the correponding error matrix. We now have a model of order 5 which is quite close to the data (Figure 5.2(c)) but the error plot (Figure 5.2(d)) shows that we are not quite there. So we choose the next measurement at 4.9 rad/sec

and obtain a sixth order model which is actually the original system we started with. The error plot in Figure 5.3(b) indeed shows very small errors.



(a) Order 6 (b) Order 6

Figure 5.3: Model of dimension 6

5.2.2 Real adaptive approach

The previous procedure can be adapted to lead to a real system. At each step, one needs to add the two measurements which correspond to the two highest errors: the first one will be used as right interpolation data, together with its complex conjugate, while the other one, together with its complex conjugate, will be used as left interpolation data.

The algorithm starts by computing a system of order N_p from N_p measurements selected from the N_s available (we assume, without loss of generality, that N_p is an even quantity; otherwise, we start by selecting $N_p + 1$ measurements). The N_p (or $N_p + 1$) indices for these measurements are linearly distributed between 1 and N_s. The system is constructed by building the Λ, M, $\mathbf{R}$, $\mathbf{L}$, $\mathbf{W}$, $\mathbf{V}$, $\mathbb{L}$ and $\sigma\mathbb{L}$ matrices as described in section 5.1.3 (with the sampling directions ℓ_i and $\mathbf{r}_i$ chosen as outlined in Eq. (5.22) and (5.23)) and setting $\mathbf{E}$ as $-\mathbb{L}$, $\mathbf{A}$ as $-\sigma\mathbb{L}$, $\mathbf{B}$ as $\mathbf{V}$, $\mathbf{C}$ as $\mathbf{W}$ and $\mathbf{D}$ as $\mathbf{0}$. Starting with a system of dimension N_p ensures that all the singular values of the intermediary model are non-zero.

At each sample frequency point, the N_p singular values of the error matrix obtained as the difference between the S-parameter data matrix and the transfer function of the current model evaluated at that particular sample are computed. The next measurements are chosen at the two frequencies where the two maximum errors (singular values) occur. At this stage we keep a record of the measurements which have already been selected and use new samples in the future.

The sampling directions for the new measurements can be chosen as

- random complex vectors (**algorithm 4**)
- the singular vectors associated with the two largest singular values of the error matrix (**algorithm 6**)
- the sum of the singular vectors of the error matrices which give the two largest errors (**algorithm 8**)

At the next step, a system of order $N_p + 2$ is constructed from the initial N_p measurements together with the two which lead to the largest errors by following the steps described in section 5.1.1. We preserve the system built up to this stage and only append the last row or column obtained from combining the new measurement with all the previous ones. The singular values of the error matrix are computed again and two new measurements are chosen according to the same criterion as before (have not already been used and give the largest error). The procedure continues until the desired accuracy or the desired order of the reduced model has been reached. Note that in this case, the algorithm constructs interpolants of even order.

5.2.3 Complex adaptive approach with block processing

We start by computing a system of order N_p from N_p measurements selected from the N_s available. The N_p indices for these measurements are linearly distributed between 1 and N_s. The system is constructed by building the Λ, M, $\mathbf{R}$, $\mathbf{L}$, $\mathbf{W}$, $\mathbf{V}$, $\mathbb{L}$ and $\sigma\mathbb{L}$

matrices as described in section 5.1.1 (with the sampling directions ℓ_i and $\mathbf{r}_i$ chosen as random complex vectors) and setting $\mathbf{E}$ as $-\mathbb{L}$, $\mathbf{A}$ as $-\sigma\mathbb{L}$, $\mathbf{B}$ as $\mathbf{V}$, $\mathbf{C}$ as $\mathbf{W}$ and $\mathbf{D}$ as $\mathbf{0}$.

At each sample frequency point, we compute the N_p singular values of the error matrices. The error matrices are computed as the difference between the scattering matrices and the transfer function of the current model evaluated at that particular frequency sample. For each one of the N_p singular values, the next measurement is chosen at the frequency where the maximum error (singular value) occurs. The ℓ_i and $\mathbf{r}_i$ sampling directions of the new measurements are chosen as random complex vectors. At this stage we keep a record of the measurements which have already been selected and use new samples in the future. Therefore, for each one of the N_p singular values, one measurement which satisfies both the condition that the error in the singular values is large and that it has not already been used is picked. This approach adaptively builds the models by adding a block of N_p measurements at a time.

At the next step, a system of order $2 \cdot N_p$ is constructed and the singular values of the error matrices are computed again. The next set of N_p measurements will be selected according to the same criterion and the procedure continues until the desired accuracy or the desired order of the reduced model have been reached.

The above procedure constitutes **algorithm 9**.

We can improve the accuracy of the model we constructed by adding another set of N_p measurements and projecting to the desired order using the singular vectors obtained from performing a singular value decomposition of a linear combination of the current Loewner and shifted Loewner matrices, for example $\sigma\mathbb{L} - j\omega_1\mathbb{L}$. This procedure constitutes **algorithm 10**.

Algorithm 11 is based on the previous one but it differs in the way the sampling directions are chosen. For the first step, namely when the first N_p measurements are

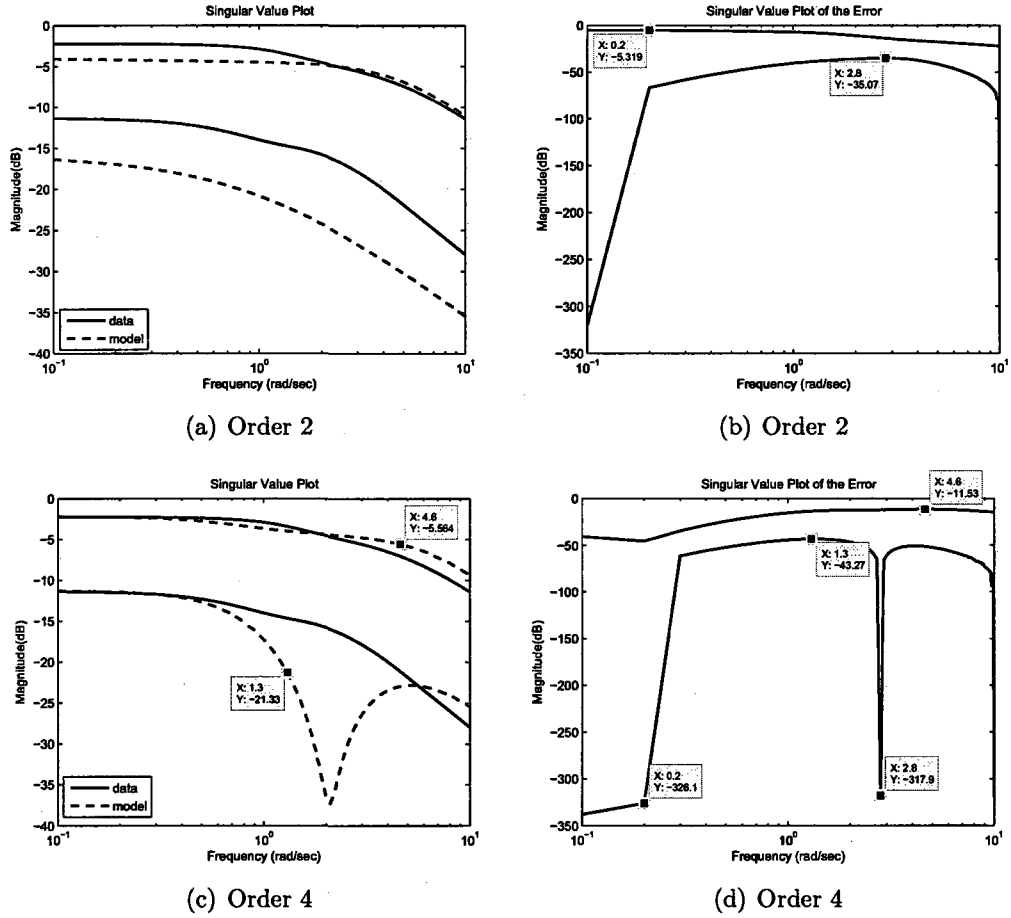
chosen, the sampling directions are taken as the sum of the singular vectors of the measured scattering matrices, as outlined in Eqs.(5.11) and (5.12). In the following steps, once we already have a model available but the order of the system is lower than desired (or the accuracy is not as low as desired), we choose the next N_p measurements which give the largest error in each of the N_p singular values of the error matrices, with the corresponding sampling directions taken as the singular vectors associated with those particular singular values, rather than random, as we had them before.

Let us make the steps of algorithm 11 more clear by considering the following example. A multiple-input multiple-output (MIMO) bounded-real system is given by the matrix transfer function

$$\mathbf{H}(s) = \begin{bmatrix} \frac{1}{2.5(s+1)} & \frac{1}{1.3[(s+1)^2+1]} \\ \frac{1}{[(s+2)^2+3^2]} & \frac{3}{s+5} \end{bmatrix}.$$

The system is of order 6 with poles at $-1 \pm j$, $-2 \pm 3j$, -1 and -5. We sample the transfer function at 100 linearly spaced frequency points between 10^{-1} rad/sec and 10^{1} rad/sec and assume that the measurements are noisy (we have available only the first 4 digits rather than the exact value of the transfer function).

We start with a system of order N_p, which is 2 in this case, obtained using the first and last measurement with the tangential directions constructed from the SVD of the scattering matrices. We plot it against the data in Figure 5.4(a). Figure 5.4(b) shows the two singular values of all the error matrices. We choose the next $N_p = 2$ measurements at the frequencies where the largest error occurs in each singular value of the error matrices occurs (0.2 for the first and 2.8 for the second singular value), with the sampling directions of the new measurements taken as the singular vectors associated with that particular singular value of the error matrix. Thus we obtain a fourth order system shown in Figure 5.4(c). The error plot in Figure 5.4(d) shows that the second singular value of the error matrices is very small at all the frequencies



(a) Order 2

(b) Order 2

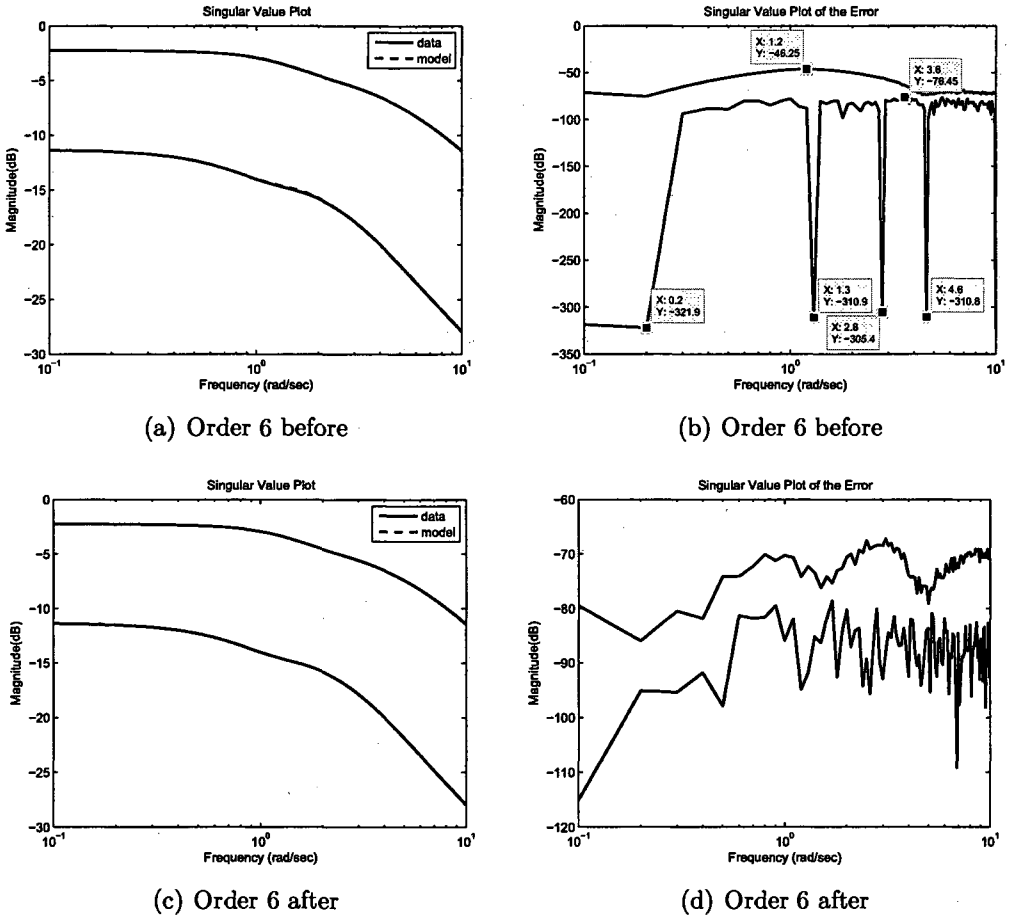
(c) Order 4

(d) Order 4

Figure 5.4: Intermediary models of order 2 and 4

which we have already used (0.1, 0.2, 2.8 and 10 rad/sec).

We choose the next measurements at 1.3 and 4.6 rad/sec, where the largest errors in each of the two singular values occurs and obtain a sixth order system shown in Figure 5.5(a). At this point we have a sixth order model for a sixth order system. However, recall that the measurements were noisy with the noise level of 10^{-4}. The error plot in Figure 5.5(b) shows that the error is below -46dB which is quite high. Therefore, in order to improve the accuracy of our models, we choose two more measurements according to the same criterion as before (at 1.2 and 3.6 rad/sec) and obtain a system of order 8. By building the projectors from the left and right singular vectors associated with the largest 6 singular values of a linear combination of the



(a) Order 6 before

(b) Order 6 before

(c) Order 6 after

(d) Order 6 after

Figure 5.5: Models of dimension 6 before and after projection

Loewner and shifted Loewner matrices of dimension 8×8, we obtain a model of order 6 which is much better than the previous one. Figure 5.5(d) shows that the maximum error is now around -70dB, much better than before.

5.2.4 Complex adaptive approach with reusing measurements

Algorithms 12, 14 and 16 are the same as the ones presented in section 5.2.1 (**Complex adaptive approach**) except for the fact that now we allow measurements to be used more than once by not keeping track of the indices which were already chosen. Note that the algorithm does not break down because the sampling directions used at this point are different from the ones used initially.

5.2.5 Real adaptive approach with reusing measurements

Algorithms 13, 15 and 17 are the same as the ones presented in section 5.2.2 (**Real adaptive approach**) except for the fact that now we allow measurements to be used more than once. Special care must be taken to ensure that, when a measurement which has already been used is to be added again, it should be used as left or right tangential data, depending on how it was previously used. If this condition is not enfored, the algorithm will break down. This is due to the fact that, when we compute the difference of the frequencies in the denominator of the Loewner matrices, we will get infinity (recall that the denominator is $\mu_i - \lambda_k$, where μ_i is the frequency used in left tangential data and λ_k is the frequency used in right tangential data). Therefore, if the same frequency is used both as left and right data, the algorithm breaks down. Note that no redundant information will be used due to the fact that the sampling directions used at this point are different from the ones used initially.

5.2.6 Complex adaptive approach with block processing and reusing measurements

Algorithms 18, 19 and 20 are the same as the ones presented in section 5.2.3 (**Complex adaptive approach with choosing N_p measurements at a time**) except for the fact that now we allow measurements to be used more than once by not keeping track of the indices which were already chosen. Note that the algorithm does not break down because the sampling directions used at this point are different from the ones used initially.

5.3 Recursive approach

The algorithms presented in this section are based on the recursive approach of building the Loewner matrix pencil and the theory presented in section 4.2.2. The advan-

tage of this approach is that we are able to identify immediately when the matrix pencil becomes singular, so one does not need to rely on the singular value decomposition or the Kronecker canonical form to identify the regular part of the system.

5.3.1 Complex recursive approach

Algorithm 21 starts by generating random complex vectors as sampling tangential directions for each measurement and collapsing each scattering matrix of dimension N_p^2 to a vector of dimension N_p. We denote these matrices by $\mathbf{L}_o$, $\mathbf{R}_o$, $\mathbf{V}_o$ and $\mathbf{W}_o$. We take the first measurement and generate an order 1 system with

$$\mathbb{L} = \frac{\mathbf{v}_{o,1}\mathbf{r}_{o,1} - \ell_{o,1}\mathbf{w}_{o,1}}{-j\omega_1 - j\omega_1}, \tag{5.41}$$

$$\sigma\mathbb{L} = \frac{-j\omega_1\mathbf{v}_{o,1}\mathbf{r}_{o,1} - j\omega_1\ell_{o,1}\mathbf{w}_{o,1}}{-j\omega_1 - j\omega_1}, \tag{5.42}$$

$$\mathbf{V} = \mathbf{v}_{o,1} \tag{5.43}$$

$$\mathbf{W} = \mathbf{w}_{o,1}, \tag{5.44}$$

where $\ell_{o,1}$ and $\mathbf{v}_{o,1}$ are the first rows of $\mathbf{L}_o$ and $\mathbf{V}_o$, while $\mathbf{r}_{o,1}$ and $\mathbf{w}_{o,1}$ are the first columns of $\mathbf{R}_o$ and $\mathbf{W}_o$, respectively. We evaluate the current generating system and its inverse at all the frequency samples and compute the errors given by:

$$\begin{bmatrix} \ell_{e,k} & \mathbf{v}_{e,k} \end{bmatrix} = \begin{bmatrix} \ell_{o,k} & \mathbf{v}_{o,k} \end{bmatrix} \Theta(\mu_k) \text{ for } k = 1, \ldots, N_s \tag{5.45}$$

$$\begin{bmatrix} -\mathbf{w}_{e,k} \\ \mathbf{r}_{e,k} \end{bmatrix} = \bar{\Theta}(\lambda_k) \begin{bmatrix} -\mathbf{w}_{o,k} \\ \mathbf{r}_{o,k} \end{bmatrix} \text{ for } k = 1, \ldots, N_s, \tag{5.46}$$

where Θ and $\bar{\Theta}$ are given by Eq. (4.17) and (4.18). We choose the next measurement for which the magnitude of the error vectors $\mathbf{v}_{e,k}$ and $\mathbf{w}_{e,k}$ is the largest and construct the new interpolant recursively using the formulas (4.59)-(4.62), where $\mathbb{L}_1$, $\sigma\mathbb{L}_1$, $\mathbf{V}_1$

and $\mathbf{W}_1$ are the matrices built up to now and

$$\mathbb{L}_2 = \frac{\mathbf{v}_{e,k}\mathbf{r}_{e,k} - \ell_{e,k}\mathbf{w}_{e,k}}{-j\omega_k - j\omega_k}, \tag{5.47}$$

$$\sigma\mathbb{L}_2 = \frac{-j\omega_k\mathbf{v}_{e,k}\mathbf{r}_{e,k} - j\omega_k\ell_{e,k}\mathbf{w}_{e,k}}{-j\omega_k - j\omega_k}, \tag{5.48}$$

$$\mathbf{V}_2 = \mathbf{v}_{e,k}, \tag{5.49}$$

$$\mathbf{W}_2 = \mathbf{w}_{e,k}, \tag{5.50}$$

where k is the index which leads to $\mathbf{v}_{e,k}$ and $\mathbf{w}_{e,k}$ having the largest norm. Using the current interpolant and the associated generating systems, the new errors are computed for all frequency measurements and the next point to be used is chosen where the largest error in both $\mathbf{v}_e$ and $\mathbf{w}_e$ occurs.

This process continues until the entry which is to be added to the Loewner matrix is close to machine precision (recall that the Loewner matrix constructed recursively is diagonal, as seen from Eq. (4.59); in other words, we stop whenever $\mathbb{L}_2$ is small enough). At this point, we distinguish two cases:

- $\sigma\mathbb{L}_2$ is small as well, so incorporating the new measurement into the current model will make the Loewner pencil numerically singular. Therefore, we stop and return the descriptor-form representation of the model with $\mathbf{E}$ as $-\mathbb{L}_1$, $\mathbf{A}$ as $-\sigma\mathbb{L}_1$, $\mathbf{B}$ as $\mathbf{V}_1$, $\mathbf{C}$ as $\mathbf{W}_1$ and $\mathbf{D}$ as $\mathbf{0}$.

- $\sigma\mathbb{L}_2$ is not small, so in this case the original system has a non-zero $\mathbf{D}$-term. We can recover the original system by building a model of dimension N_p more. Therefore, we need to add N_p more measurements to the already chosen ones. This can be achieved by computing the errors $\mathbf{v}_{e,k}$, $\ell_{e,k}$, $\mathbf{r}_{e,k}$ and $\mathbf{w}_{e,k}$ at all the N_s samples, sorting the $\mathbf{v}_{e,k}$ vectors according to their norm in descending order and choosing the corresponding N_p measurements for which the errors are the largest. We build the $\mathbb{L}_2$, $\sigma\mathbb{L}_2$, $\mathbf{V}_2$ and $\mathbf{W}_2$ matrices from these N_p frequency

points and construct the recursive $\mathbb{L}$, $\sigma\mathbb{L}$, $\mathbf{V}$ and $\mathbf{W}$ according to Eq. (4.59)-(4.62). The resulting Loewner matrix will be singular, with the dimension of the null-space equal to N_p, while the resulting shifted Loewner matrix will be non-singular. In this construction, the model will have, apart from the original poles, N_p infinite poles.

5.3.2 Real recursive approach

Algorithm 22 is a modified version of the previous one to yield real systems with poles coming in complex conjugate pairs by adding, at each step, two measurements instead of one. One of the measurements, together with its complex conjugate will be used as left tangential data, while the other one will be used as right tangential data.

This algorithm starts by splitting the measurements in two, assuming that N_s is an even number: the first half of the measurements, together with their complex conjugates, will be used as right data, while the second half and their complex conjugates, as left data. For each frequency sample, random sampling tangential directions are generated and the scattering matrix of dimension N_p^2 is collapsed to a vector of dimension N_p. We denote these matrices by $\mathbf{L}_o$, $\mathbf{R}_o$, $\mathbf{V}_o$ and $\mathbf{W}_o$.

We take the first measurement and its conjugate, together with the $\frac{N_s}{2}+1$ measurement, and generate an order 2 system:

$$\mathbb{L}_{i,k} = \frac{\mathbf{v}_{o,i}\mathbf{r}_{o,k} - \ell_{o,i}\mathbf{w}_{o,k}}{\mu_i - \lambda_k},\ i,k = 1,2, \tag{5.51}$$

$$\sigma\mathbb{L}_{i,k} = \frac{\mu_i\mathbf{v}_{o,i}\mathbf{r}_{o,k} - \lambda_k\ell_{o,i}\mathbf{w}_{o,k}}{\mu_i - \lambda_k}\ i,k = 1,2, \tag{5.52}$$

$$\mathbf{V} = \begin{bmatrix} \mathbf{v}_{o,1} \\ \bar{\mathbf{v}}_{o,1} \end{bmatrix} \tag{5.53}$$

$$\mathbf{W} = \begin{bmatrix} \mathbf{w}_{o,1} & \bar{\mathbf{w}}_{o,1} \end{bmatrix}, \tag{5.54}$$

where $\mathbf{v}_{o,2} = \bar{\mathbf{v}}_{o,1}$, $\mathbf{r}_{o,2} = \bar{\mathbf{r}}_{o,1}$, $\ell_{o,2} = \bar{\ell}_{o,1}$, $\mathbf{w}_{o,2} = \bar{\mathbf{w}}_{o,1}$, $\lambda_1 = j\omega_1$, $\lambda_2 = -j\omega_1$, $\mu_1 = j\omega_{\frac{N_s}{2}+1}$, $\mu_2 = -j\omega_{\frac{N_s}{2}+1}$. We evaluate the current generating system and its inverse at all the frequency samples and compute the errors as in Eq. (5.45) and (5.46). We choose the next measurement with the largest magnitude of the error vectors $\mathbf{w}_{e,k}$ and use it, together with its complex conjugate, as right data, while the measurement with the largest magnitude of the error vector $\mathbf{v}_{e,k}$ is used, together with its complex conjugate, as left data. Next, we construct the new model recursively using the formulas (4.59)-(4.62), where $\mathbb{L}_1$, $\sigma\mathbb{L}_1$, $\mathbf{V}_1$ and $\mathbf{W}_1$ are the matrices built up to now and $\mathbb{L}_2$, $\sigma\mathbb{L}_2$, $\mathbf{V}_2$, $\mathbf{W}_2$ are constructed as in Eqs.(5.51)-(5.54) from the two measurements which yield the largest errors. Using the current interpolant and the associated generating systems, the new errors are computed for all frequency measurements and the next two frequency points are chosen where the largest errors in $\mathbf{v}_e$ and $\mathbf{w}_e$ occur.

This process continues until the entries which are to be added to the Loewner matrix are close to machine precision (recall that the Loewner matrix constructed recursively is block-diagonal, as seen from Eq. (4.59); in other words, we stop whenever the norm of $\mathbb{L}_2$ is small). At this point, we distinguish two cases:

- the norm of $\sigma\mathbb{L}_2$ is small as well, so incorporating the new measurement into the current model will make the Loewner pencil numerically singular. Therefore, we stop and return the descriptor-form representation of the model with $\mathbf{E}$ as $-\mathbb{L}_1$, $\mathbf{A}$ as $-\sigma\mathbb{L}_1$, $\mathbf{B}$ as $\mathbf{V}_1$, $\mathbf{C}$ as $\mathbf{W}_1$ and $\mathbf{D}$ as $\mathbf{0}$.

- the norm of $\sigma\mathbb{L}_2$ is not small, so in this case the original system has a non-zero $\mathbf{D}$-term. We can recover the original system by building a model of dimension N_p more. Therefore, we need to add N_p more measurements to the already chosen ones. This can be achieved by computing the errors $\mathbf{v}_{e,k}$, $\ell_{e,k}$, $\mathbf{r}_{e,k}$ and $\mathbf{w}_{e,k}$ at all the N_s samples, sorting the $\mathbf{v}_{e,k}$ vectors according to their norm in descending order and choosing the corresponding N_p measurements for which

the errors are the largest. We build the $\mathbb{L}_2$, $\sigma\mathbb{L}_2$, $\mathbf{V}_2$ and $\mathbf{W}_2$ matrices from these N_p frequency points and construct the recursive $\mathbb{L}$, $\sigma\mathbb{L}$, $\mathbf{V}$ and $\mathbf{W}$ according to Eq. (4.59)-(4.62). The resulting Loewner matrix will be singular, with the dimension of the null-space equal to N_p, while the resulting shifted Loewner matrix will be non-singular. In this construction, the model will have, apart from the original poles, N_p infinite poles.

Clearly, this algorithm will generate interpolants of even order.

5.3.3 Complex recursive approach with reusing measurements

Algorithm 23 is similar to the one presented in section 5.3.1 except for the following remark: assuming we have found that the measurement at position k leads to the largest magnitude in the error vectors $\mathbf{v}_{e,k}$ and $\mathbf{w}_{e,k}$, we generate a new random vector $\mathbf{r}_{o,k}$, set $\ell_{o,k} = \mathbf{r}_{o,k}^*$, compute $\mathbf{v}_{o,k} = \ell_{o,k}\tilde{\mathbf{S}}_k$ and $\mathbf{w}_{o,k} = \mathbf{S}_k\mathbf{r}_{o,k}$ and update the old values. With this approach, measurements may be selected more than once, but the information used at this step is different from before due to the fact that the sampling directions we use are different.

5.3.4 Real recursive approach with reusing measurements

Algorithm 24 is similar to the one presented in section 5.3.2 except for the following remark: assuming we have found that the measurements at positions k_1 and k_2 lead to the largest magniture in the error vectors $\mathbf{v}_{e,k_1}$ and $\mathbf{w}_{e,k_2}$, we generate new random vectors ℓ_{o,k_1} and $\mathbf{r}_{o,k_2}$, compute the new $\mathbf{v}_{o,k_1}$ and $\mathbf{w}_{o,k_2}$ and update the old values. Also, at positions $k_1 + 1$ and $k_2 + 1$, we need to store their complex conjugate values. With this approach, measurements may be selected more than once, but the information used this time is different from before due to the fact that the sampling directions are different.

Remark. Given an appropriate accuracy of fit or desired order of the macromodel, the resulting system will be stable and passive, provided that the measurements come from a stable and passive device. Often however, due to measurements errors, the data is not passive (i.e. the maximum of the largest singular value is bigger than one). In this case, the resulting macromodel may not passive. In case the maximum of the largest singular value is slightly above one (i.e. by 10% or less), the easiest way to obtain a passive model is to divide either the $\mathbf{B}$ or the $\mathbf{C}$ state-space matrix of the corresponding realization by that maximum. Another option for a posteriori passivation would be to apply the passivity enforcement described in [12].

CHAPTER 6

Computational Complexity

6.1 Vector fitting

The computational complexity of the column-wise vector fitting algorithm is

$$O\left(N_1 N_p N^2 N_s + N_2 N_p^4 N^2 N_s\right),$$

where the order k of the resulting system is a multiple of the number of ports ($k = N_p \cdot N$), N is the number of starting poles, N_1 is the number of iterations used to fit the column sum of each column and last, N_2 is the number of iterations used to fit each column. The first term in the complexity comes from solving a least-squares problem with an $\mathbf{A}$ matrix of dimension $N_s \times 2N$ when fitting the column sum of each column using N_1 iterations. The second term in the complexity is due to the fact that a least-squares problem with an $\mathbf{A}$ matrix of dimension $(N_s N_p) \times (n(N_p + 1))$ needs to be solved when fitting each column using N_2 iterations.

If matrix-wise vector fitting would have been used, the dimension of the resulting models would be a multiple of the number of ports N_p, but the computational complexity would scale with the 6-th power of N_p. On the other hand, the complexity of element-wise fitting scales with the 2-nd power of N_p, but the order of the resulting

models will be too large, as they are multiples of N_p^2. Therefore, column-wise vector fitting provides the best trade-off between the dimension of the macromodel and the computational complexity, for the case of devices with a large number of ports.

The complexity dependence of the column-wise vector fitting algorithm on the 4-th power of N_p clearly shows that, as the number of ports increases, vector fitting becomes more computationally expensive, so it would be unfeasible for devices with the number of ports in the order of hundreds.

6.2 Singular value decomposition approach

Analyzing algorithm 1 from the point of view of the computational complexity involved reveals that it is

$$O\left(N_s N_p^2 + N_s^2 N_p + N_s^3\right),$$

for the case when random sampling directions are used.

The first term comes from creating the tangential data, as each of the N_s scattering matrices of dimension N_p^2 is projected to one vector of dimension N_p. The second term in the complexity comes from computing the Loewner and shifted Loewner matrices: each of its N_s^2 entries is computed as the difference between two inner products of vectors of dimension N_p. Last, the N_s^3 term comes from computing the singular value decomposition of a linear combination of the Loewner and shifted Loewner matrices.

As for the case when the sampling directions are chosen as the sum of the singular vectors of the scattering matrices, the computational complexity is

$$O\left(N_s N_p^3 + N_s^2 N_p + N_s^3\right).$$

The only difference from before is in the first term, which comes from creating the tangential data: for each of the N_s scattering matrices of dimension N_p^2, we compute

its singular value decomposition.

The computational complexity for algorithm 2 is the same as for algorithm 1. However, when implemented in Matlab, creating the Loewner matrices takes longer in the real approach, but performing the singular value decomposition of the Loewner pencil is faster due to the fact that matrix operations are faster on real-valued matrices.

6.3 Adaptive approach

6.3.1 Complex adaptive approach

The computational complexity of **algorithms 3, 5 and 7** is

$$O\left(N_s k^4 + N_s N_p^3 k + k N_s \log(N_s)\right),$$

where k is the desired order of the reduced system, or the dimension of the model which meets the desired error criterion.

Creating the N_p order system from the first N_p measurements is of complexity $O\left(N_p^3\right)$. Next we evaluate the transfer function (for which we need to invert a matrix of dimension m) for all dimensions of the models between N_p and k at all frequency samples:

$$O\left(N_s \sum_{m=N_p}^{k} m^3\right) = O\left(N_s \left[\left(\frac{k(k+1)}{2}\right)^2 - \left(\frac{(N_p-1)N_p}{2}\right)^2\right]\right) \tag{6.1}$$

$$= O\left(N_s k^4\right) \tag{6.2}$$

Following, we compute the singular value decomposition of all N_s error matrices,

which are of dimension N_p^2, for all dimensions of the models between N_p and k:

$$O\left(N_s \sum_{m=N_p}^{k} N_p^3\right) = O\left(N_s N_p^3 (k - N_p + 1)\right) = O\left(N_s N_p^3 k\right). \tag{6.3}$$

Afterwards we sort a vector of dimension N_s containing, for each sample, the largest singular value of the error matrix. This is performed for each dimension of the model between N_p and k:

$$O\left(\sum_{m=N_p}^{k} N_s \log(N_s)\right) = O\left(k N_s \log(N_s)\right). \tag{6.4}$$

Last, we create the tangential data for each new measurement which is added, as the scattering matrix of dimension N_p^2 is projected to one vector of dimension N_p. This is performed for all dimensions of the model between N_p and k:

$$O\left(\sum_{m=N_p}^{k-1} N_p^2\right) = O\left(k N_p^2\right). \tag{6.5}$$

6.3.2 Real adaptive approach

The computational complexity of **algorithms 4, 6 and 8** is the same as that of algorithms 3, 5 and 7. However, due to the fact that in the real adaptive approach we add two frequency points at each step, the constants are half those in the complex case, so the computational time is also half.

6.3.3 Complex adaptive approach with block processing

The order of the models constructed with **algorithms 9, 10 and 11** is a multiple of the number of ports ($k = N_p \cdot N$). Their computational complexity is

$$O\left(N N_p N_s \log(N_s) + N_s N_p^3 N^4\right).$$

Creating the N_p order system from the first N_p measurements is of complexity $O\left(N_p^3\right)$. Next, for each frequency sample, we evaluate the transfer function for all models of dimension m , where m is a multiple of the number of ports N_p ($m = h \cdot N_p$):

$$O\left(N_s \sum_{h=1}^{N} (h \cdot N_p)^3\right) = O\left(N_s N_p^3 \sum_{h=1}^{N} h^3\right) \tag{6.6}$$

$$= O\left(N_s N_p^3 N^4\right). \tag{6.7}$$

Following, we compute the singular value decomposition of all N_s error matrices, which are of dimension N_p^2, for all models of dimension m, where m is a multiple of the number of ports N_p ($m = h \cdot N_p$):

$$O\left(N_s \sum_{h=1}^{N} N_p^3\right) = O\left(N_s N_p^3 N\right). \tag{6.8}$$

Afterwards, we sort N_p vectors of dimension N_s, each vector containing, for each frequency sample, one of the singular values of the error matrix. This is performed for each dimension of the intermediary model which is a multiple of N_p, between N_p and $N \cdot N_p$:

$$O\left(N_p \sum_{h=1}^{N} N_s \log(N_s)\right) = O\left(N_p N N_s \log(N_s)\right). \tag{6.9}$$

Next we create the tangential data for all new N_p measurements which are added at a time, as the scattering matrices of dimension N_p^2 are projected to one vector of dimension N_p. This is performed for all dimensions of the model which are multiples of N_p, between N_p and $N \cdot N_p$:

$$O\left(\sum_{h=1}^{N} N_p N_p^2\right) = O\left(N N_p^3\right). \tag{6.10}$$

Last, we create the models of dimensions which are multiples of N_p, between N_p and

$N \cdot N_p$:

$$O\left(\sum_{m=N_p \cdot 1}^{N \cdot N_p} m^2 N_p\right) = O\left(N_p^3 \sum_{h=1}^{N} h^2\right) = O\left(N_p^3 N^3\right). \tag{6.11}$$

6.3.4 Complex adaptive approach with reusing measurements

The computational complexity of **algorithms 12, 14 and 16** is

$$O\left(N_s k^4 + N_s N_p^3 k\right),$$

where k is the desired order of the reduced system, or the dimension of the model which meets the desired error criterion. The expression is similar to the one for the algorithms in the complex adaptive approach (algorithms 3, 5 and 7). However, we do not need to sort the vector of dimension N_s containing the largest singular values of the error matrices, but only find the maximum of that vector. This is due to the fact that we are allowed to reuse measurements.

6.3.5 Real adaptive approach with reusing measurements

The computational complexity for **algorithms 13, 15 and 17** is the same as for algorithms in the complex adaptive approach with multiple use of measurements (algorithms 12, 14 and 16). However, due to the fact that in the real approach we add two frequency points at each step, the constants are half those in the complex case, so usually, the computational time is also half.

6.3.6 Complex adaptive approach with block processing and reusing measurements

The order of the models which are constructed with **algorithms 18, 19 and 20** is a multiple of the number of ports ($k = N_p \cdot N$) and their computational complexity is

$$O\left(N_s N_p^3 N^4\right).$$

The expression is similar to the one for the algorithms in the complex adaptive approach with choosing N_p measurements at a time. However, we do not need to sort the N_p vectors of dimension N_s containing the singular values of the error matrices, but only find the maximum of each vector. This is due to the fact that we are allowed to reuse measurements.

6.4 Recursive Approach

The computational complexity of all the algorithms in the recursive approach is

$$O\left(N_s k^4 + N_s N_p k^3 + N_s N_p^2 k^2\right),$$

where k is the desired order of the reduced system. The first step is creating the tangential data with random sampling directions. As each of the N_s scattering matrices of dimension N_p^2 is projected to a vector of dimension N_p, this yields a complexity of

$$O\left(N_s N_p^2\right).$$

The next step is evaluating the generating system at all the frequency samples. This operation is performed for each intermediary model of dimension m between 1 and

k, so the complexity is:

$$O\left(N_s \sum_{m=1}^{k} \left(m^3 + N_p m^2 + N_p^2 m\right)\right) = O\left(N_s k^4 + N_s N_p k^3 + N_s N_p^2 k^2\right). \quad (6.12)$$

The next step consists of computing the norm of all the error vectors $\mathbf{v}_{e,k}$ and $\mathbf{w}_{e,k}$, $k = 1, \ldots, N_s$, which are of dimension N_p, and selecting the one with the largest norm. This operation is performed for each intermediary model of dimension m between 1 and k, so the complexity is:

$$O\left(N_s N_p \sum_{m=1}^{k} m\right) = O\left(N_s N_p k^2\right). \quad (6.13)$$

The next step consists of creating the order $m+1$ model, by appending the last row and column to the already available order m model, for $m = 1, \ldots, k-1$:

$$O\left(N_p \sum_{m=1}^{k-1} m\right) = O\left(N_p k^2\right). \quad (6.14)$$

6.5 Comparison of the algorithms in terms of computational complexity

Let us summarize the computational complexity of all the algorithms proposed and column-wise vector fitting. For the purpose of comparison, we assume that the dimension of the model is a multiple of the number of ports N_p: $k = N \cdot N_p$.

We conclude that the algorithms which are based on block processing (9-11 and 18-20) are the least computationally expensive due to the fact that their complexity scales with the third power of the number of ports and with N_s or $N_s \log(N_s)$ in the number of samples. As we will see from chapter 7, these algorithms are also accurate, as they are able to construct good models in a short time.

Algorithm	Complexity
VF	$O\left(N_s N_p N^2 N_1 + N_s N_p^4 N^2 N_2\right)$
SVD approach with random sampling directions (1, 2)	$O\left(N_s N_p^2 + N_s^2 N_p + N_s^3\right)$
SVD approach using the SVD of the scattering matrices (1, 2)	$O\left(N_s N_p^3 + N_s^2 N_p + N_s^3\right)$
complex and real adaptive (4-8)	$O\left(N_s N_p^4 N^4 + N_s \log(N_s) N_p N\right)$
complex adaptive with block processing (9-11)	$O\left(N_s \log(N_s) N_p N + N_s N_p^3 N^4\right)$
complex and real adaptive with reusing measurements (12-17)	$O\left(N_s N_p^4 N^4\right)$
complex adaptive with block processing and reusing measurements (18-20)	$O\left(N_s N_p^3 N^4\right)$
complex and real recursive (21-24)	$O\left(N_s N_p^4 N^4\right)$

Table 6.1: Computational Complexity of VF and the algorithms proposed; N_s denotes the number of samples of the data set, N_1 and N_2 are the number of iterations used in the pole-relocation process of vector fitting

CHAPTER 7

Numerical Examples

This chapter will analyze a theoretical example, as well as two examples obtained from measurements. More examples are found in Appendix B. Our analysis will include all the algorithms proposed in the previous chapter, as well as the state-of-the-art *vector fitting* method and will focus on the accuracy of the macromodels and on the CPU time required to produce such a model. Models of the same dimension were constructed using all the algorithms and the singular values of the one which provided the best trade-off between accuracy and computational time was plotted against the singular values of the measured S-parameters. Moreover, the small deviations in the singular values were proven to be an indication of the good quality of the model by comparing the magnitude and angle of some (or all) of the individual entries of the S-parameters of the original data to our model and to the model obtained with VF.

Because of the motivation underlying the whole area of model order reduction (namely that large macromodels are expensive to use in simulations), we are interested in determining systems of the least possible order which satisfy an a priori given error criterion, or which model the data accurately in the frequency band of interest.

The accuracy and quality of the models was assessed using two error measures:

- the normalized $\mathcal{H}_\infty$-norm of the error system, which is the largest singular value

of the difference between the data and the model evaluated at all the frequency points divided by the largest singular value of the data:

$$\mathcal{H}_\infty \text{ error } = \frac{\max_{i=1\ldots N_s} \sigma_1 \left(\mathbf{S}_i - \mathbf{H}(j\omega_i)\right)}{\max_{i=1\ldots N_s} \sigma_1 \left(\mathbf{S}_i\right)},$$

where $\sigma_1(\cdot)$ denotes the largest singular value of $(\cdot)$.

- the normalized $\mathcal{H}_2$-norm of the error system:

$$\mathcal{H}_2 \text{ error } = \frac{\sum_{i=1}^{N_s} \|\mathbf{S}_i - \mathbf{H}(j\omega_i)\|_F^2}{\sum_{i=1}^{N_s} \|\mathbf{S}_i\|_F^2}$$

where $\|\cdot\|_F^2$ stands for the Frobenius-norm of the respective matrix, namely the sum of the magnitude squared of all N_p^2 entries, which, for the error matrix, is:

$$\|\mathbf{S}_i - \mathbf{H}(j\omega_i)\|_F^2 = \sum_{k_1=1}^{N_p} \sum_{k_2=1}^{N_p} |\mathbf{S}_{k_1,k_2,i} - \mathbf{H}_{k_1,k_2}(j\omega_i)|^2 .$$

The first error measure, namely the normalized $\mathcal{H}_\infty$-norm, evaluates the maximum deviation in the singular values, while the second measure, namely the normalized $\mathcal{H}_2$-norm, evaluates the error in the magnitude of all entries, proving to be a good estimate of the overall performance.

All experiments which involved column-wise vector fitting used the same options:

- the starting poles are real and stable, linearly distributed in the frequency band
- for each of the N_p columns, a new set of poles is obtained after fitting the column sum of the respective column with $N_1 = 5$ iterations
- for each of the N_p columns, the set of starting poles found by fitting the column sum is used to fit the respective column with $N_2 = 5$ iterations

- the model was not required to produce any **D** or **E** matrix as in Eq. (3.1), unless otherwise specified

- since it was shown in [24] that relaxed vector fitting has better properties in terms of pole relocation and, consequently, produces better models, our simulations used vector fitting without the asymptotic condition on the scaling function $\sigma(s)$ to approach unity at high frequencies, but with a relaxed constraint introduced to obtain a nontrivial solution for the least-squares problem.

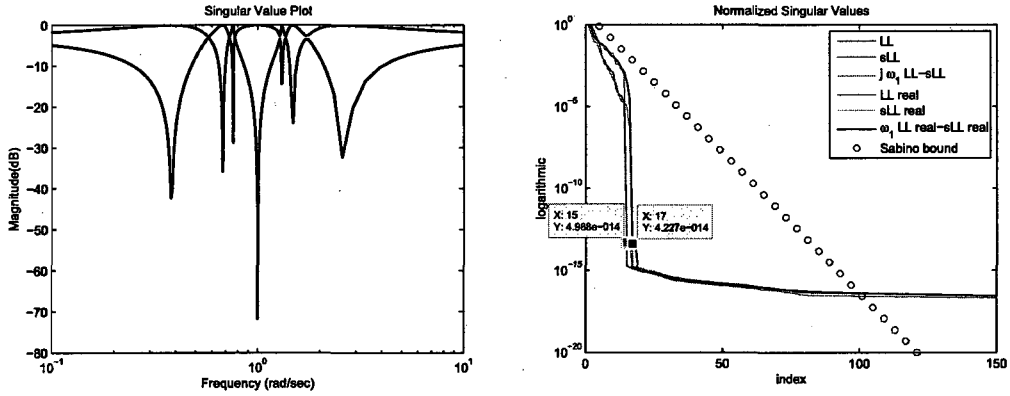
- the input weight vector was computed from the measurements as the inverse of the magnitude function and individual weighting was employed for each entry of the matrix.

Recall that column-wise vector fitting produces systems of orders which are multiples of the number of ports. In the numerical examples presented below, we have chosen the dimension of the resulting macromodel as a multiple of the number of ports. This is not necessary for our algorithms; its sole purpose is to be able to compare our results with those obtained with vector fitting.

The experiments were performed on a Pentium Dual-Core at 2.2GHz with 3GB RAM.

7.1 Noise-free system with 2 ports, 14 poles and non-zero D matrix

We will consider a theoretical system of order 14 with $N_p = 2$ ports. Unlike the previous examples which were analyzed, this system has a non-zero D-term. We would like to compare all the algorithms previously proposed with vector fitting in terms of computational time and normalized errors when trying to recover the original system from 616 noise-free measurements. We sample the transfer function between 10^{-1} rad/sec and 10^{1} rad/sec.



(a) Original system (b) Singular value drop

Figure 7.1: Original system and singular value drop of the Loewner matrix pencil

Fig. 7.1(a) shows the sigma plot, while Fig. 7.1(b) shows the plot of the normalized singular values of the Loewner and shifted Loewner matrices constructed using the complex as well as the real approach, with the sampling tangential directions taken as outlined in Eq. (5.11), (5.12), (5.11) and (5.12). In the last figure, only the first 150 singular values, of all the 616 ones, are shown. The singular values which are not shown are smaller than machine precision. The red circles show the Sabino bound, which predicts that the dimension of the regular part of the Loewner pair is 100, much larger than the true order, 16.

We are able to bypass the introducion of a D-term in our model by generating

systems of order 16 which have a singular **E**-matrix and an invertible **A** matrix, resulting in 2 infinite eigenvalues, together with the other 14 poles of the original system. The plot of the singular values clearly shows that the singular values of the Loewner matrix contructed using either the complex or the real approach drop below machine precision at the 15-th one, while the singular values of a linear combination between the Loewner and the shifted Loewner matrix constructed using both approaches drop below machine precision at the 17-th one. This is due to the fact that the shifted Loewner matrix itself is of rank 16.

Applying the GUPTRI software to the Loewner pencil constructed using the complex approach reveals the fact that the finite part is of dimension 14, the dimension of the singular part is $600 = 600 - 0 = 602 - 2$ and there are $2 = 2 - 0$ infinite eigenvalues. The original poles are accurately recovered as the eigenvalues of the matrix sub-pencil given by the first 14 rows and the columns 601 to $616 - 2 = 614$.

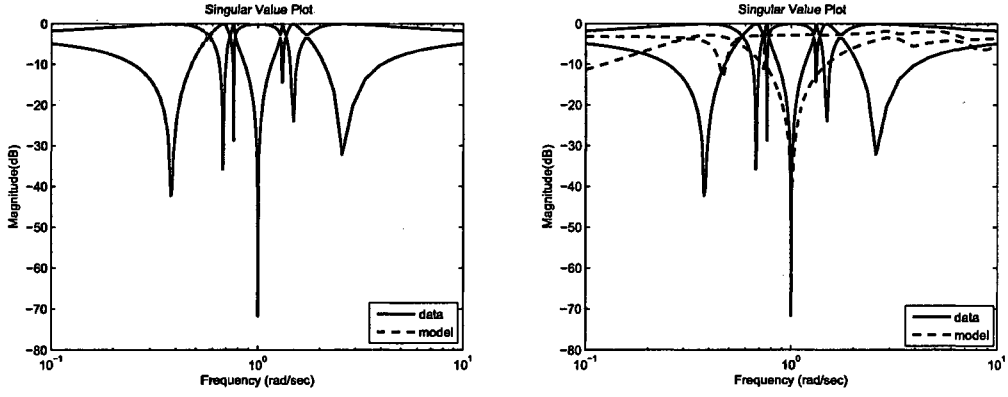
```
>> [Sc,Tc,Pc,Qc,kstrc]=guptri(sLL,LL,eps);
>> kstrc
kstrc =
   600    -1   602     0    -1    14    -1
     0    -1     2     0    -1    14    -1
>> eig(Sc(1:14,601:614),Tc(1:14,601:614))
ans =
 -1.8847e-01 + 5.5712e-01i
 -1.8847e-01 - 5.5712e-01i
 -3.9862e-02 + 6.9312e-01i
 -3.9862e-02 - 6.9312e-01i
 -6.1213e-03 - 7.6186e-01i
 -6.1213e-03 + 7.6186e-01i
 -8.7744e-01 - 4.7969e-01i
 -8.7744e-01 + 4.7969e-01i
 -1.0545e-02 + 1.3125e+00i
 -1.0545e-02 - 1.3125e+00i
 -8.2699e-02 - 1.4380e+00i
 -8.2699e-02 + 1.4380e+00i
 -5.4486e-01 - 1.6106e+00i
 -5.4486e-01 + 1.6106e+00i
```

Using the real approach of constructing the matrices, we obtain the same output

as for the complex approach.

```
>> [Sr,Tr,Pr,Qr,kstrr]=guptri(sLLN,LLN,eps);
>> kstrr
kstrr =
   600    -1   602     0    -1    14    -1
     0    -1     2     0    -1    14    -1
>> eig(Sr(1:14,601:614),Tr(1:14,601:614))
 -1.8847e-01 + 5.5712e-01i
 -1.8847e-01 - 5.5712e-01i
 -3.9862e-02 + 6.9312e-01i
 -3.9862e-02 - 6.9312e-01i
 -6.1213e-03 - 7.6186e-01i
 -6.1213e-03 + 7.6186e-01i
 -8.7744e-01 - 4.7969e-01i
 -8.7744e-01 + 4.7969e-01i
 -1.0545e-02 + 1.3125e+00i
 -1.0545e-02 - 1.3125e+00i
 -8.2699e-02 - 1.4380e+00i
 -8.2699e-02 + 1.4380e+00i
 -5.4486e-01 - 1.6106e+00i
 -5.4486e-01 + 1.6106e+00i
```

For this example, each of our algorithms constructed an order 16 model, while vector fitting was given $N = 7$ starting poles, so the state-space realization was of dimension 14, and was required to produce a $\mathbf{D}$ matrix.



(a) $k = 16$ system with algorithm 8 (b) $k = 14$ system with VF

Figure 7.2: Models obtained with algorithm 8 and with VF for a data set with $N_p = 2$ ports

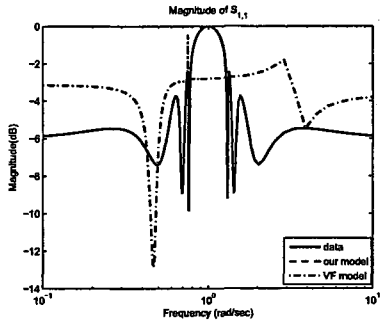
Table 7.1 presents the CPU time required by each algorithm to build the desired

Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	6.115	5.2562e-011	1.0733e-022
1 with SVD	7.316	7.9168e-011	5.0374e-022
1 with GUPTRI	71.1	2.0233e-009	1.6335e-020
2 with random	6.786	1.5458e-012	1.6292e-026
2 with SVD	6.021	1.5858e-012	2.0590e-026
2 with GUPTRI	78.01	1.9437e-012	4.0002e-026
3	0.811	2.8079e-009	1.0534e-019
4	0.546	4.3503e-011	3.9535e-023
5	0.686	5.1074e-007	2.1425e-015
6	0.421	4.2601e-012	3.4000e-025
7	0.717	2.9360e-007	5.4628e-016
8	**0.358**	**4.3167e-013**	**2.0011e-026**
9	0.436	5.7950e-009	5.4103e-019
10	0.468	1.4195e-009	4.6871e-020
11	0.514	2.8908e-010	8.8401e-022
12	0.671	1.1175e-007	9.7231e-017
13	0.546	1.4541e-012	1.2452e-025
14	0.733	3.9186e-007	1.3394e-015
15	0.483	6.9265e-011	7.9872e-023
16	0.686	2.6723e-007	4.3479e-016
17	0.499	3.1271e-012	2.0392e-025
18	0.421	1.5337e-009	3.7054e-020
19	0.499	2.7260e-009	2.7162e-019
20	0.514	8.4259e-010	9.6214e-021
21	1.731	3.5326e-010	1.2717e-021
22	4.992	1.0450e-009	1.4614e-020
23	1.638	2.7428e-008	7.0880e-018
24	5.772	1.0352e-008	1.1177e-018
VF	**0.639**	**3.0409e-001**	**9.0143e-001**

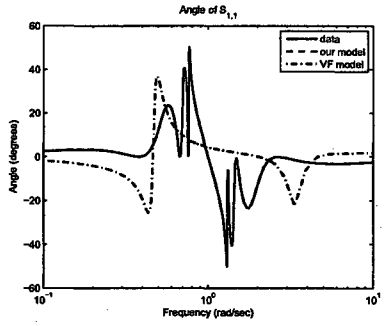
Table 7.1: Results for $N_s = 616$ noise-free measurements of an order 14 MIMO system with $N_p = 2$ ports

model, as well as the normalized $\mathcal{H}_\infty$ and $\mathcal{H}_2$ errors for the resulting systems. We conclude that all the algorithms proposed were able to recover the original system, yielding small errors. The fastest algorithm was algorithm 8, which constructed the best model in terms of the resulting errors in under 0.4s.

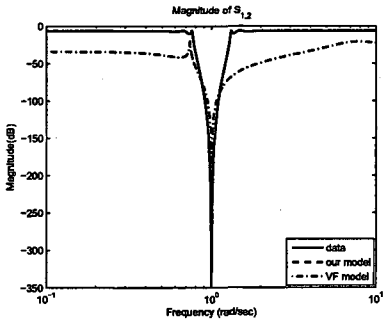
The singular value plots of the model built with algorithm 8 and the model built with VF are shown in Figure 7.2. Vector fitting clearly shows deviations, while our model has perfectly reconstructed the original system.



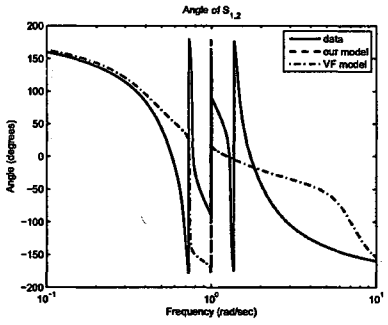
(a) Magnitude of $S_{1,1}$



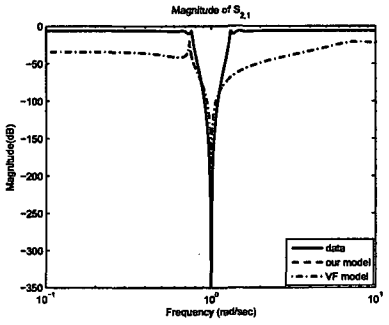
(b) Angle of $S_{1,1}$



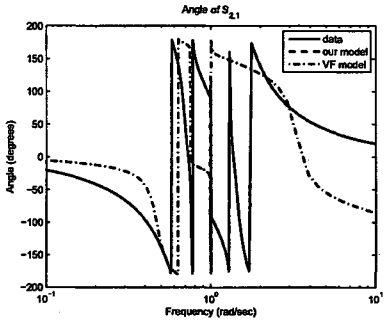
(c) Magnitude of $S_{1,2}$



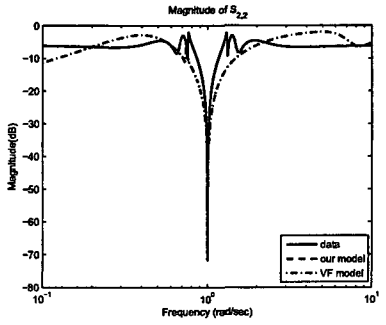
(d) Angle of $S_{1,2}$



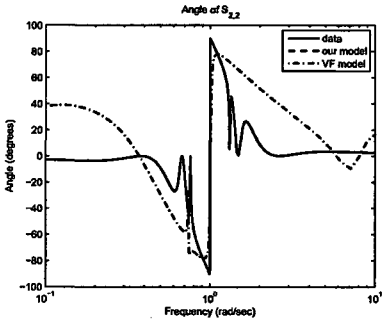
(e) Magnitude of $S_{2,1}$



(f) Angle of $S_{2,1}$



(g) Magnitude of $S_{2,2}$



(h) Angle of $S_{2,2}$

Figure 7.3: Comparing algorithm 8 and VF to the data

Remark. Vector fitting does NOT recover the original system. One needs to go up to order 200 in order to have perfect match on the plot, with errors of $1.4736e-001$ for the $\mathcal{H}_\infty$ and $2.1447e-004$ for the $\mathcal{H}_2$. The errors are still high, showing that the original 14-th order system is not recovered, but only interpolated!

We also checked how well the magnitude and the angle of the entries are modeled. Figure 7.3 compares the measured $S_{1,1}$, $S_{1,2}$, $S_{2,1}$ and $S_{2,2}$ entries to the model obtained with algorithm 8 and to the one obtained with VF. Clearly, our model is not distinguishable from the data, while VF is far from good.

7.2 Examples obtained from measurements

The data sets were provided by CST Ltd. They were obtained as measurements taken with a vector network analyzer (VNA). There are N_s frequency samples. For each frequency sample a matrix of dimension $N_p \times N_p$ with complex entries, representing the measured S-parameters, is given.

7.2.1 100 measurements from a device with 50 ports

This data set contains $N_s = 100$ frequency samples between 10MHz and 1GHz. In order to avoid numerical instabilities, all frequencies were scaled by 10^{-6}. Note that for systems in descriptor form, the inverse of this scaling factor can be included in the **E**-matrix, while for systems in state-space form, the scaling factor should be included in the **A**-matrix and the inverse of the scaling factor should be included in the **B** or **C** matrix.

Clearly, the pencil constructed by using the sampling directions as outlined in Eq. (5.11), (5.12), (5.22) and (5.23) is not singular, since the singular values drop only to 10^{-10}, as seen in Figure 7.4. However, the Sabino bound predicts no drop between the first and the last singular value (the red circle in the top right corner comes from

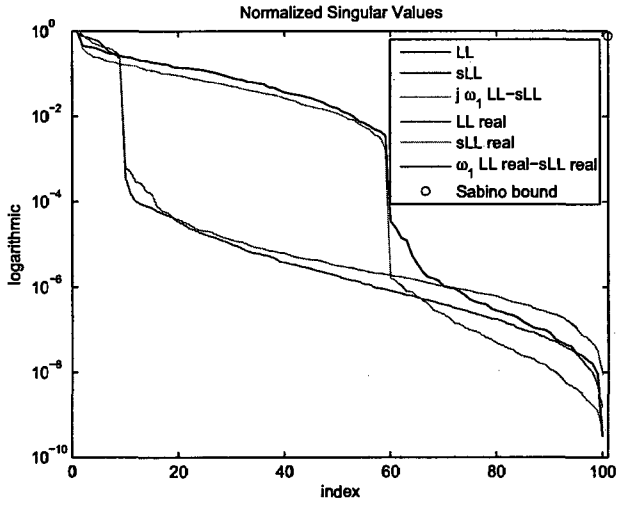


Figure 7.4: Singular value drop of the Loewner pencil

Sabino's bound).

The fact that the pencil is regular is also confirmed by GUPTRI, which gives no singular part:

```
>> [Sc,Tc,Pc,Qc,kstrc]=guptri(sLL,LL,10^(-13));
>> kstrc
kstrc =
    -1    -1   100    -1
    -1    -1   100    -1
>> [Sr,Tr,Pr,Qr,kstrr]=guptri(sLLN,LLN,10^(-13));
>> kstrr
kstrr =
    -1    -1   100    -1
    -1    -1   100    -1
```

Due to the fact that, to compare models obtained with our algorithms to VF we need to choose the order of the interpolating system as a multiple of the number of ports, we are presenting the results for interpolants of degree $k = 100$ (choosing $k = 50$ would lead to bad interpolants) in Table 7.2 and Figure 7.5. The x-axis of the plots in Figure 7.5 have a logarithmic scale and the frequencies are scaled by 10^{-6}.

Table 7.2 clearly shows that all the algorithms proposed yielded better interpolants than VF. Some of the computational times were too large, even though the errors

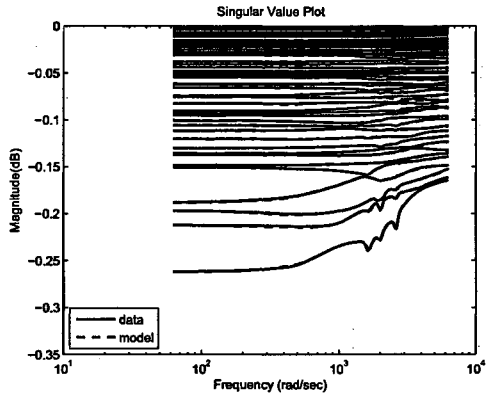
Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	0.109	1.8549e-001	2.7800e-005
1 with SVD	0.9828	1.9494e-002	3.1844e-006
2 with random	0.124	2.0641	2.2276e-003
2 with SVD	1.404	4.0389e-001	4.3264e-005
3	17.94	1.4566e-001	1.1067e-005
4	10.28	1.0685e-001	6.9278e-006
5	19.251	7.2000e-002	3.1008e-006
6	9.0169	2.6766e-002	2.3139e-007
7	18.361	6.6265e-002	3.1466e-006
8	9.5941	3.9146e-002	3.5263e-006
9	1.3572	7.3338e-002	2.3371e-006
10	1.0764	8.5047e-002	5.7580e-006
11	1.6848	1.6385e-001	4.0373e-005
12	17.987	3.2658e-002	3.9051e-006
13	9.5785	1.0188e-001	1.5273e-005
14	18.907	2.8043e-002	3.5697e-007
15	**10.514**	**6.1068e-003**	**1.9976e-007**
16	18.861	1.0216e-001	3.9970e-006
17	10.717	4.7962e-001	5.9538e-005
18	1.326	9.5822e-002	1.0360e-005
19	2.262	2.7269e-002	2.5591e-006
20	**3.1824**	**1.8681e-002**	**6.0492e-007**
21	67.486	8.6151e-002	9.0725e-006
22	21.715	7.0267e-002	3.6118e-006
23	33.259	1.8143e-002	1.2337e-006
24	11.076	3.8795e-002	5.2383e-006
VF	**6.1152**	**8.1173e-001**	**2.0818e-002**

Table 7.2: Results for constructing a model of dimension $k = 100$ from a data set obtained from a device with $N_p = 50$ ports

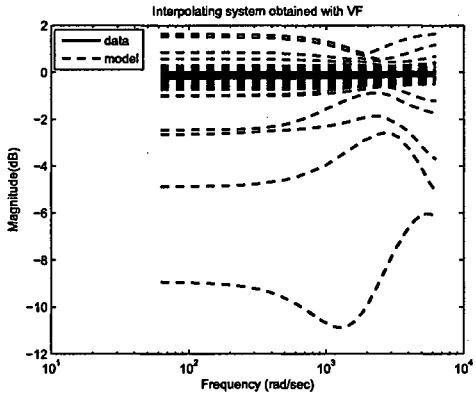
were small. Overall, we conclude that algorithms 1 with SVD, 20, 19, 18, 9, 10 constructed a good model in less than 3.5s, while algorithm 15 built the best model, but the computational time was large.

Vector fitting starts improving its performance, so for an order $k = 300$ model, the errors are comparable to our algorithms. However, the computational time has also increased to 18.252s.

We also checked how well the magnitude and the angle of some of the entries are modeled. Figure 7.6 compares the measured $S_{1,1}$ and $S_{10,20}$ entries to the model

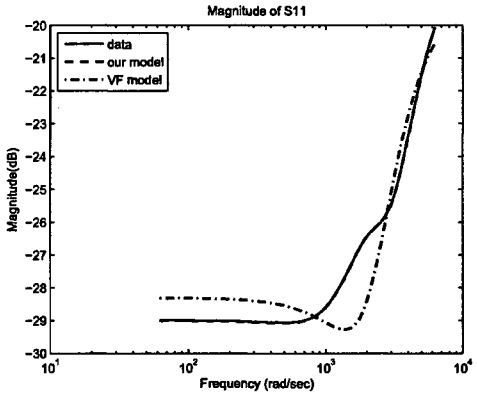


(a) $k = 100$ system with algorithm 15

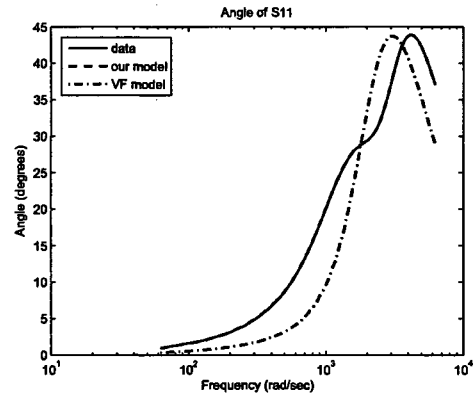


(b) $k = 100$ system with VF

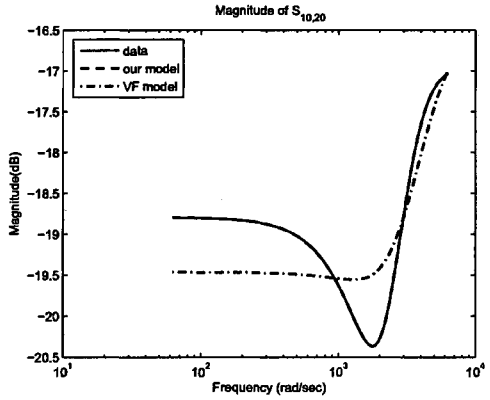
Figure 7.5: Models of dimension $k = 100$ obtained with algorithm 15 and with VF for a data set with $N_p = 50$ ports



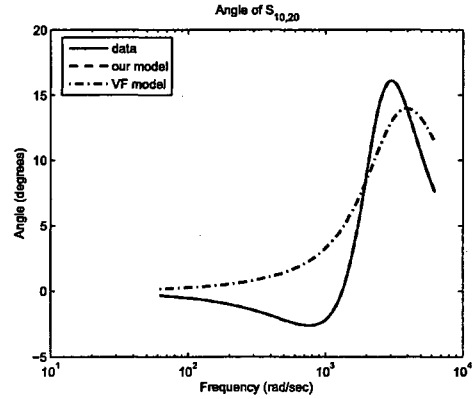
(a) Magnitude of $S_{1,1}$



(b) Angle of $S_{1,1}$



(c) Magnitude of $S_{10,20}$



(d) Angle of $S_{10,20}$

Figure 7.6: Comparison of the models built with algorithm 15 and with VF to the data obtained from a device with $N_p = 50$ ports

obtained with algorithm 15 and to the one obtained with VF. Clearly, our model is hardly distinguishable from the data, while VF is far from good.

7.2.2 1000 measurements from a device with 14 ports

This data set contains $N_s = 1000$ frequency samples between 6MHz and 6GHz. In order to avoid numerical instabilities, all frequencies were scaled by 10^{-9}.

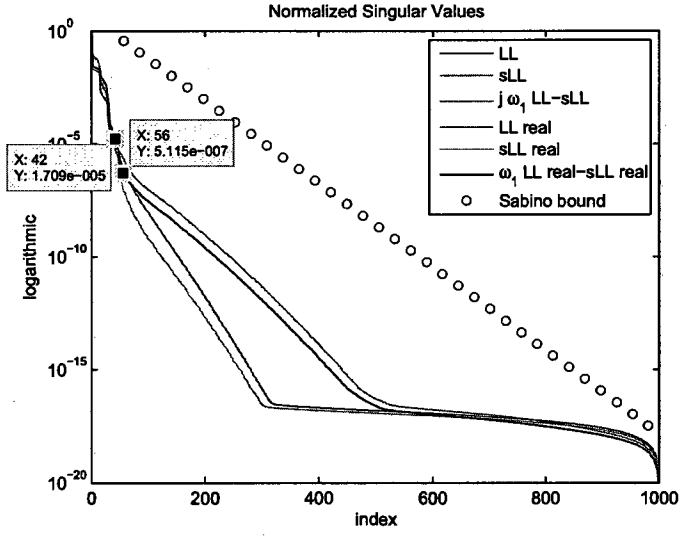


Figure 7.7: Singular value drop of the Loewner pencil

First, we construct the big Loewner and shifted Loewner matrices in the real and complex approach using random sampling directions of the left and right tangential interpolation data. Next, we compute how fast the singular values drop and check whether Sabino's bound is able to match the drop of the normalized singular values. Figure 7.7 shows that the predicted decay is much slower than the actual decay.

The GUPTRI software takes very long to run on such big matrices (the matrices are of dimension 1000) and we could not obtain an answer even after 2h.

Due to the fact that, to compare models obtained with our algorithms to VF we need to choose the order of the interpolating system as a multiple of the number of ports, we are presenting the results for interpolants of degree $k = 42$ (choosing $k = 14$

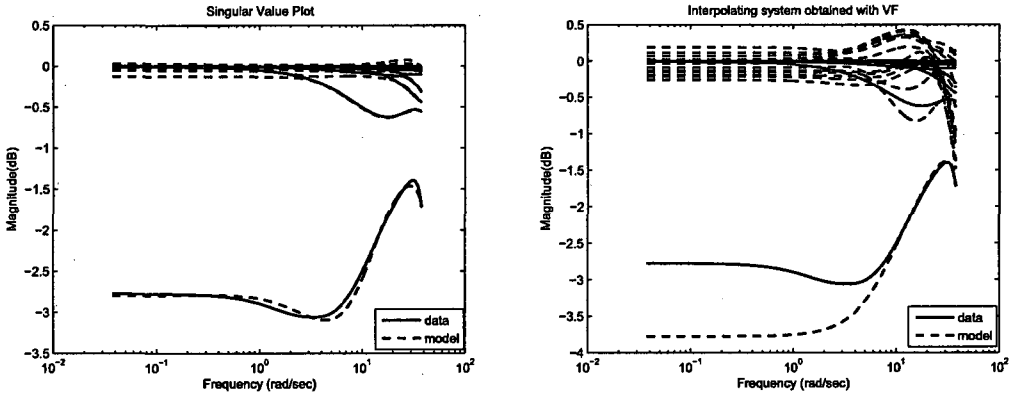
or $k = 14 \cdot 2 = 28$ would lead to bad interpolants) in Table 7.3 and Figure 7.8.

Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	26.271	7.1572e-002	3.5793e-005
1 with SVD	21.731	8.5170e-002	8.3270e-005
2 with random	**26.941**	**5.1305e-002**	**2.0354e-005**
2 with SVD	16.162	1.0884e-001	1.3526e-004
3	11.404	1.1858e-001	7.4909e-004
4	6.505	4.8471e-001	6.8743e-003
5	11.357	3.6150e-002	8.0072e-005
6	6.084	1.4840e-001	4.8100e-004
7	11.06	9.4299e-002	5.4434e-004
8	6.037	3.5106e-001	5.5071e-003
9	1.45	7.1743e-001	1.2331e-002
10	2.355	1.3782e-001	2.4533e-004
11	2.184	4.0158e-002	7.4119e-005
12	11.31	2.6106e-001	2.7126e-003
13	6.567	10.445	8.9309e-002
14	11.232	2.8553e-002	5.4404e-005
15	5.959	2.5826e-001	1.0119e-003
16	10.998	1.5608e-001	4.2956e-004
17	6.162	3.7347e-001	3.7533e-003
18	1.419	2.5424e-001	1.7114e-003
19	2.106	5.3735e-002	8.3176e-005
20	2.23	3.0314e-002	5.9915e-005
21	33.041	2.1006e-001	8.0776e-004
22	31.996	2.0001	8.9327e-003
23	33.915	1.9271e-001	5.5677e-004
24	31.715	11.429	6.8304e-001
VF	**7.238**	**1.9702e-001**	**2.4979e-003**

Table 7.3: Results for constructing a model of dimension $k = 42$ from a data set obtained from a device with $N_p = 14$ ports

Clearly, even though some of the algorithms proposed yielded good models, the computational times were too large (for example, algorithms 1 and 2, the latter giving the best order $k = 42$ model when the sampling directions are chosen outlined in Eq. (5.22) and (5.23), as well as algorithms 5 and 14, with a computational time of less than 11.5s). Overall, we conclude that algorithms 11, 19, 20 provide the best accuracy for a reasonable computational time (less than 2.5s).

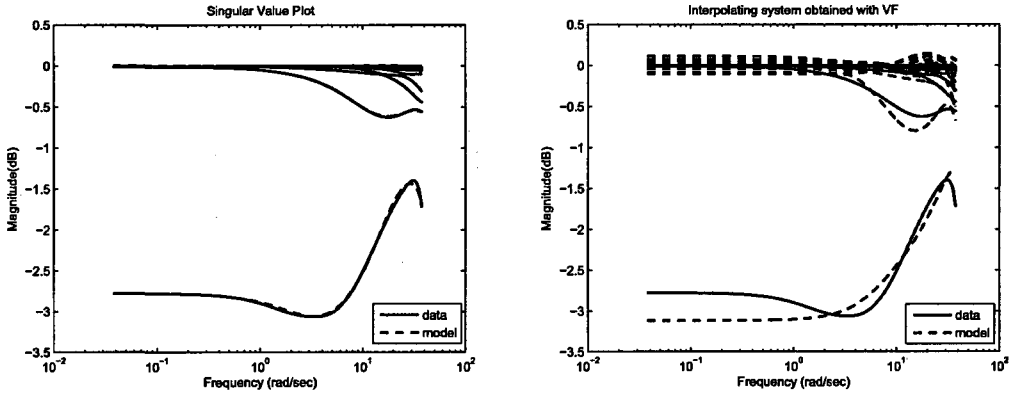
Next, let us compare the computational time and the errors when an order $k = 56$



(a) $k = 42$ system with algorithm 20 (b) $k = 42$ system with VF

Figure 7.8: Models of dimension $k = 42$ obtained with algorithm 20 and with VF for a data set with $N_p = 14$ ports

model is generated using all the algorithms. The results are presented in Figure 7.9 and Table 7.4.



(a) $k = 56$ system with algorithm 11 (b) $k = 56$ system with VF

Figure 7.9: Models of dimension $k = 56$ obtained with algorithm 11 and with VF for a data set with $N_p = 14$ ports

Clearly, even though some of the algorithms proposed yielded good models, the computational times were too large (for example, algorithms 1,2, 5 and 7, with 2 giving the best order $k = 56$ model). Overall, we conclude that algorithms 10, 11, 19, 20 provide the best accuracy for a reasonable computational time (less than 4s).

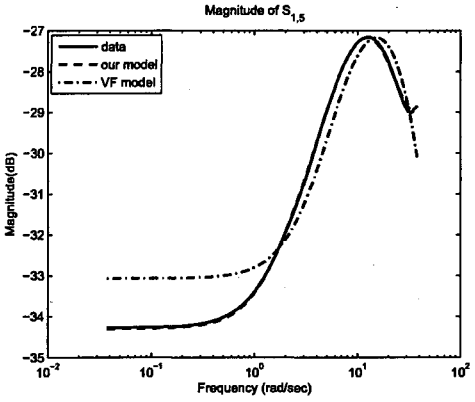
We believe that we have already found a good model for the data, so we do not

Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	24.679	2.3784e-002	2.2634e-006
1 with SVD	22.495	2.3003e-001	2.2025e-004
2 with random	**24.867**	**1.0988e-002**	**7.2808e-007**
2 with SVD	17.035	3.0181e-002	9.0578e-006
3	23.213	3.3664e-002	2.1821e-005
4	11.887	1.8921e-001	8.4886e-004
5	21.747	3.0019e-002	5.3164e-006
6	11.076	7.2923e-002	5.5111e-005
7	21.871	1.4180e-002	5.7364e-006
8	11.965	9.4900e-002	1.4807e-004
9	2.199	7.7044e-002	8.9283e-005
10	3.603	2.5243e-002	5.2952e-006
11	**3.759**	**9.8189e-003**	**4.0007e-006**
12	23.697	7.5533e-002	1.0875e-004
13	12.043	2.0851e-001	1.2738e-003
14	22.355	2.7307e-002	1.3730e-005
15	12.168	8.2653e-002	2.3920e-004
16	22.308	4.8226e-001	1.5954e-003
17	11.528	4.6674e-001	1.4649e-003
18	2.121	2.2391e-002	1.6510e-005
19	3.65	1.1810e-002	2.8685e-006
20	3.369	1.1105e-002	4.8149e-006
21	65.957	1.078	7.4311e-003
22	40.217	1.9228e-001	1.8773e-004
23	67.548	1.8851e-001	1.4508e-003
24	39.312	8.4512e-002	4.9230e-005
VF	**10.234**	**7.9651e-002**	**2.5404e-004**

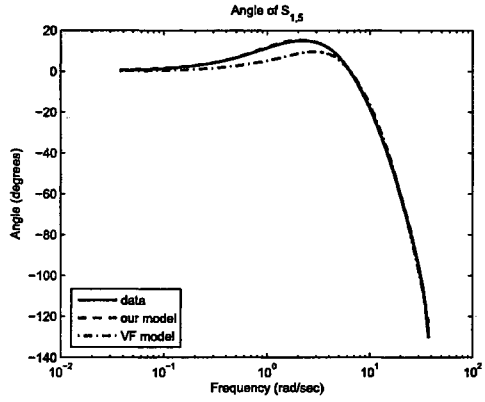
Table 7.4: Results for constructing a model of dimension $k = 56$ from a data set obtained from a device with $N_p = 14$ ports

need to search any further. Increasing the order of the desired interpolant would decrease the errors, but will also increase the computational time.

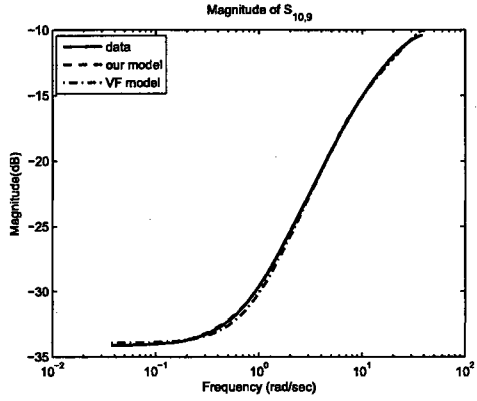
We also checked how well the magnitude and the angle of some of the entries are modeled. Figure 7.10 compares the measured $S_{1,5}$ and $S_{10,9}$ entries to the model obtained with algorithm 11 and to the one obtained with VF. Clearly, our model is hardly distinguishable from the data, while VF shows small deviations.



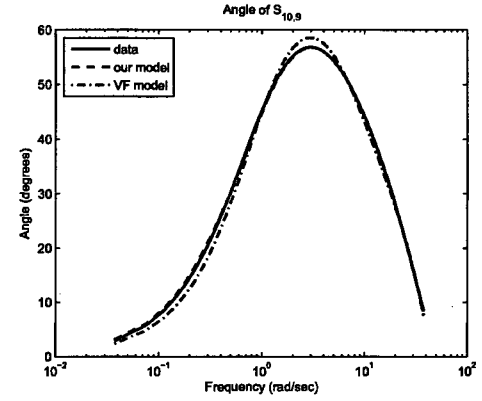
(a) Magnitude of $S_{1,5}$



(b) Angle of $S_{1,5}$



(c) Magnitude of $S_{10,9}$



(d) Angle of $S_{10,9}$

Figure 7.10: Comparison of the models built with algorithm 11 and with VF to the data obtained from a device with $N_p = 14$ ports

CHAPTER 8

Conclusion and Future Work

The computational complexity of the algorithms proposed depends upon three parameters:

- the number of ports of the device (i.e., clearly, the larger the number of ports, the more expensive the algorithm)
- the number of samples provided in the data set (i.e., clearly, the more samples we have, the more time is required to process the information; however, the reason behind having more measurements than needed is that more information is provided, so the results are expected to be more accurate than in the case of less measurements)
- the order of the macromodel k (i.e., clearly, the higher the dimension of the desired model, the more expensive it is to generate it, but the approximation errors will also be smaller due to the trade-off between the resulting accuracy and the complexity of the system).

However, we assume that we have no insight into the underlying properties and characteristics of the systems we are modelling, so our analysis should not include the dimension k of the macromodel.

Usually, one will perform as many measurements as is feasible, but for a large number of ports one cannot take arbitrarily many meansurements due to storage limitations (the files containing the measurements will become extremely large). Therefore, some of our algorithms are suited for data sets with a large number of samples, but a small number of ports, while others are suited for devices with a large number of ports, but not so many samples.

The algorithms suited for data sets containing a large number of samples (more than 500) but few ports (less than 15) are those which have a factor of N_s or $N_s \log(N_s)$ in the complexity: algorithms 3-24, as well as VF. Based on the examples we analyzed in chapter 7, we conclude that algorithms 14 (complex adaptive approach with sampling directions chosen as the singular vectors associated with the largest singular value of the error matrix, reusing measurements) and 5 (complex adaptive approach with sampling directions chosen as the singular vectors associated with the largest singular value of the error matrix) constructed very good models, but the computational time was large. Even though the computational complexity scales with N_s^3, algorithms 1 (singular value decomposition of the Loewner matrix pencil in the complex approach) and 2 (singular value decomposition of the Loewner matrix pencil in the real approach) were able to build good models, but, as expected, the CPU time was very large. Overall, the algorithms based on the complex adaptive approach with block processing (10-11 and 19-20) are the most efficient, based on the CPU time required to produce models with good accuracy.

However, the main focus of this work is on the case when we have a large number of ports. The algorithms suited for data sets with many ports (greater than 14) but not so many samples (less than 500) are those which have a factor of N_p^2 or N_p^3 in the complexity: 1 and 2 (with the sampling directions taken either as random complex vectors, or as the sum of singular vectors of the scattering matrices), 9-11, 18-20. Based on the examples we analyzed in chapter 7, we conclude that algorithms 1,

2, 10-11, 19-20 are the most efficient, based on the CPU time required to produce models with good accuracy.

Figure 8 presents a flow chart which summarizes last two paragraphs, indicating which algorithms to use depending on the number of ports of the device and the number of samples in the data set.

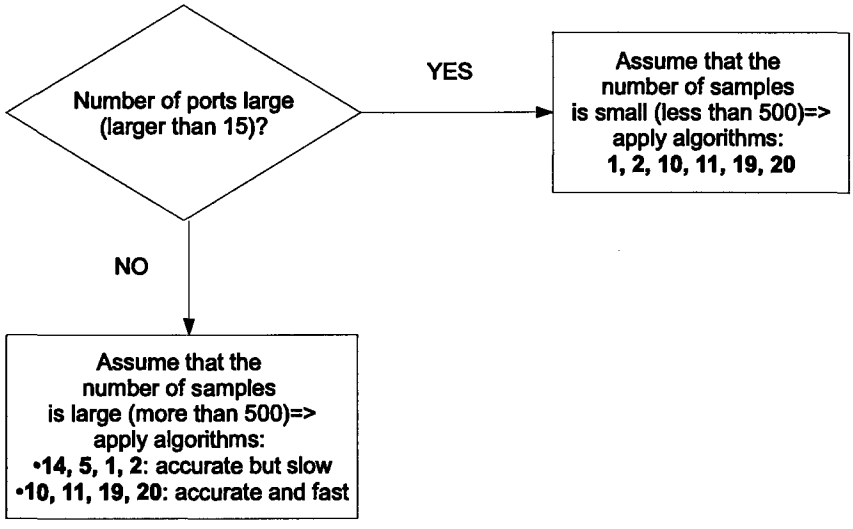


Figure 8.1: Flow chart

We conclude that, overall, algorithms **10 (complex adaptive approach with block processing and collapsing at the last step, using random sampling directions)**, **11 (complex adaptive approach with block processing and collapsing at the last step, having the sampling directions as the singular vectors associated with the largest singular value of the error matrix)**, **19 (complex adaptive approach with block processing and collapsing at the last step, using random sampling directions and reusing measurements)** and **20 (complex adaptive approach with block processing and collapsing at the last step, having the sampling directions as the singular vectors associated with the largest singular value of the error matrix, reusing measurements)** are both efficient, from the point of view of the CPU time, and accurate, from the

point of view of the resulting errors. These algorithms yield good models irrespective of the number of ports of the underlying device or the number of samples provided.

This thesis proposes several algorithms which were developed based on the system-theoretic concept of the Loewner matrix pencil and their performance was compared to the state-of-the-art vector fitting algorithm which has become the industry standard due to its relatively accurate models computed in a relatively short time. However, as mentioned in section 3, there are still some open problems with respect to this algorithm. As shown in [24] and [10], vector fitting is esentially a robust reformulation of the Sanathanan-Koerner iteration using rational basis functions instead of polynomials and pole relocation instead of weighting.

Nevertheless, this does not give any insight into the convergence properties of the iteration process used to relocate the poles of the model, starting from a set of starting poles. The poles which are given as input greatly influence the convergence rate and the accuracy of the resulting macromodels. Every application or data set may have different convergence rates even though the starting poles were chosen according to the same criterion. It was found **experimentally** that a good way of choosing the poles is either real numbers linearly distributed in the desired frequency range, or complex conjugate pairs with the imaginary part linearly distributed in the desired frequency range and the real part as 1% of the imaginary part. If this choice proves inefficient, one may have to go through a trial and error process before a better set of starting poles is found.

Moreover, there is another step in the vector fitting algorithm which lacks a formal proof. That is the process of flipping the unstable poles which result during the iteration process into the left-half plane. As before, this step has **experimentally** proven effective, but there is little insight into why this works.

As seen from the theoretical examples analyzed in sections 7.1 and B.5, vector fitting is not able to recover the original system even in the case of noise-free

measurements. Additionally, the examples coming from measurements analyzed in subsections 7.2.1-B.6.2 show that vector fitting overestimates the dimension of the macromodel, as one needs to construct models of dimension even 10 times bigger than the order of our systems in order to obtain comparable errors.

Our algorithms do not use any kind of heuristics, but only the data available from the measurements taken from the device. They require no knowledge of the underlying linear system in terms of dimension or location of poles as they construct models only by carefully arranging the matrix-measurements in a certain way. Moreover, in the case of noise-free measurements, one can determine the dimension of the original system by ploting the singular values of the Loewner pencil constructed using all the measurements available and checking at what index the first singular value falls below machine precision. Alternatively, the dimension of the underlying system can be found by computing the Kronecker canonical form of the singular Loewner pencil. However, the recursive approach is able to identify the dimension of the underlying system immediately from the norm of the Loewner and shifted Loewner matrices which are to be added. Therefore, if the norms of the Loewner matrices corresponding to the measurements which are to be incorporated are small, then those additions will make the pencil singular and, therefore, should not be included.

There are still some aspects of this thesis which will be addressed in future work:

- improve the computational time of the algorithms developed using the recursive approach (algorithms 21-24). With the current implementation, they are unfeasible due to the large CPU time required

- use the fact that in the adaptive and recursive approach the new matrices are obtained from the previous ones by appending the last row and/or column. One can make use of this fact and change the implementation of the algorithms to make them more efficient, by employing optimized algorithms dealing with rank-1 updates

- conduct Monte-Carlo simulations for the algorithms in which the sampling directions are chosen as random and average the computational time and the errors of a large number of simulations
- have the algorithms available online.

References

[1] A. C. Antoulas, *Approximation of Large-Scale Dynamical Systems.* Philadelphia: SIAM, 2005. 1, 8

[2] M. Elzinga, K. L. Virga, and J. L. Prince, "Improved global rational approximation macromodeling algorithm for networks characterized by frequency-sampled data," *IEEE Trans. Microw. Theory Tech.*, vol. 48, no. 9, pp. 1461–1468, Sep. 2000. 2

[3] R. Achar and M. Nakhla, "Efficient transient simulation of embedded subnetworks characterized by S-parameters in the presence of nonlinear elements," *IEEE Trans. Microw. Theory Tech.*, vol. 46, no. 12, pp. 2356–2363, Dec. 1998. 2

[4] T. Palenius, "Efficient time-domain simulation of interconnect characterized by large RLC circuits or tabulated S-parameters," Ph.D. dissertation, Helsinki University of Technology, Nov. 2004. 2

[5] P. V. Overschee and B. D. Moor, "Continuous-time frequency domain subspace system identification," *Signal Processing*, vol. 52, no. 2, pp. 179–194, Jul. 1996. 2

[6] F. Ebert and T. Stykel, "Rational interpolation, minimal realization and model reduction," DFG Research Center MATHEON, Tech. Rep. 371-2007, 2007. 2, 10, 14

[7] B. Gustavsen and A. Semlyen, "Rational approximation of frequency domain responses by vector fitting," *IEEE Trans. Power Del.*, vol. 14, pp. 1052–1061, Jul. 1999. 2, 10, 14, 15

[8] ——, "A robust approach for system identification in the frequency domain," *IEEE Trans. Power Del.*, vol. 19, pp. 1167–1173, Jul. 2004. 2, 10

[9] D. Deschrijver and T. Dhaene, "Passivity-based sample selection and adaptive vector fitting algorithm for pole-residue modeling of sparse frequency-domain data," in *Proc. IEEE International Conference on Behavioral Modeling and Simulation (BMAS'04)*, Oct. 2004, pp. 68–73. 2

[10] D. Deschrijver, "Broadband macromodeling of linear systems by vector fitting," Ph.D. dissertation, Universiteit Antwerpen, Oct. 2007. 3, 10, 92

[11] S. Grivet-Talocia and M. Bandinu, "Improving the convergence of vector fitting for equivalent circuit extraction from noisy frequency responses," *IEEE Trans. Electromagn. Compat.*, vol. 48, pp. 104–120, Feb. 2006. 10, 14

[12] S. Grivet-Talocia, "Passivity enforcement via perturbation of Hamiltonian matrices," *IEEE Trans. Circuits Syst. I*, vol. 51, pp. 1755–1769, Sep. 2004. 15, 62

[13] C. Schröder and T. Stykel, "Passivation of LTI systems," DFG Research Center MATHEON, Tech. Rep. 368-2007, 2007. 15

[14] C. P. Coelho, J. R. Phillips, and L. M. Silveira, "Optimization-based passive constrained fitting," in *Proc. IEEE/ACM International Conference on Computer-Aided Design(ICCAD'02)*, 2002, pp. 775–780. 15

[15] ——, "A convex programming approach to positive real rational approximation," in *Proc. IEEE/ACM International Conference on Computer-Aided Design(ICCAD'01)*, 2001, pp. 245–251. 15

[16] ——, "A convex programming approach for generating guaranteed passive approximations to tabulated frequency-data," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 23, pp. 293–301, Feb. 2004. 15

[17] B. Gustavsen and A. Semlyen, "Enforcing passivity for admittance matrices approximated by rational functions," *IEEE Trans. Power Syst.*, vol. 16, no. 1, pp. 97–104, Feb. 2001. 15

[18] A. J. Mayo and A. C. Antoulas, "A framework for the solution of the generalized realization problem," *Linear Algebra and Its Applications*, vol. 405, pp. 634–662, 2007. 17, 19, 36

[19] A. Antoulas, J. Ball, J. Kang, and J. Willems, "On the solution of the minimal rational interpolation problem," *Linear Algebra and Its Applications*, vol. 137/138, pp. 511–573, 1990. 21

[20] S. Lefteriu and A. C. Antoulas, "Interpolation of scattering parameters," Rice University, Tech. Rep. TREE0708, Jul. 2007. 37

[21] J. Sabino, "Solution of large-scale lyapunov equations via the block modified Smith method," Ph.D. dissertation, Rice University, Houston, TX, 2006. 41

[22] J. Demmel and B. Kågström, "The generalized Schur decomposition of an arbitrary pencil $A - zB$: Robust software with error bounds and applications. Part I: Theory and algorithms," *ACM Trans. Math. Softw.*, vol. 19, no. 2, pp. 160–174, 1993. 44, 45

[23] ——, "The generalized Schur decomposition of an arbitrary pencil $A - zB$: Robust software with error bounds and applications. Part II: Software and applications," *ACM Trans. Math. Softw.*, vol. 19, no. 2, pp. 175–201, 1993. 44, 45

[24] B. Gustavsen, "Improving the pole relocation properties of vector fitting," *IEEE Trans. Power Del.*, vol. 21, no. 3, pp. 1587–1592, Jul. 2006. 74, 92

[25] D. Ioan and G. Ciuprina, *Reduced Order Models of ON-chip Passive Components and Interconnects, Workbench and Test Structures*, W. Schilders and H. van der Vorst, Eds. Springer Series, 2008. 174

[26] T. G. Wright, "EigTool," 2002, software available at `http://www.comlab.ox.ac.uk/projects/pseudospectra/eigtool`. 130

APPENDIX A

Pseudocode for the algorithms

Algorithm 1 Singular value decomposition of the Loewner matrix pencil in the complex approach.

1: From the data, build the matrices Λ, M, $\mathbf{R}$, $\mathbf{W}$, $\mathbf{L}$, $\mathbf{V}$, $\mathbb{L}$ and $\sigma\mathbb{L}$ as described in section 5.1.1.
2: Plot the singular values of $\sigma\mathbb{L} - \lambda\mathbb{L}$, where λ is chosen as any frequency sample, and decide for a dimension of the reduced model based on the drop in singular values.
3: Determine the projectors $\mathbf{X}$ and $\mathbf{Y}$ from (5.14).
4: Compute the corresponding state-space matrices using the formulas in (5.15).
5: Compute the errors and decide whether the current interpolant meets the desired accuracy criterion. If not, choose a larger k and repeat steps 3 to 5.

Algorithm 2 Singular value decomposition of the Loewner matrix pencil in the real approach.

1: From the data, build the matrices Λ, M, $\mathbf{R}$, $\mathbf{W}$, $\mathbf{L}$, $\mathbf{V}$, $\mathbb{L}$ and $\sigma\mathbb{L}$ as described in section 5.1.3.
2: Plot the singular values of $\sigma\mathbb{L} - \lambda\mathbb{L}$, where λ is chosen as any frequency sample, and decide for a dimension of the reduced model based on the drop in singular values.
3: Determine the projectors $\mathbf{X}$ and $\mathbf{Y}$ from (5.14).
4: Compute the corresponding state-space matrices using the formulas in (5.25).
5: Compute the errors and decide whether the current interpolant meets the desired accuracy criterion. If not, choose a larger k and repeat steps 3 to 5.

Algorithm 3 Complex adaptive approach with random sampling directions.

1: Generate a set of N_p indices linearly distributed between 1 and N_s.
2: Set $m \leftarrow N_p$.
3: Construct the order m system by building the associated state-space matrices as described in section 5.1.1.
4: **for** $i \leftarrow 1 \ldots N_s$ **do**
5: Evaluate the transfer function at the sample point $j\omega_i$.
6: Compute the singular values of the difference between the data matrix at point ω_i and the transfer function evaluated at $j\omega_i$ and keep only the largest.
7: **end for**
8: Build a vector of indices which have already been used.
9: **while** m less than k, the desired order of the reduced system **do**
10: Select the measurement which has not been used and gives the largest error in singular values over all N_s measurements and update $m \leftarrow m + 1$, as well as the vector of already used indices.
11: Construct the order m system as described in section 5.1.1, by preserving the system built at the previous step and appending the last column and row, obtained from combining the new measurement with all the previous ones.
12: Repeat steps 4 - 7.
13: **end while**

Algorithm 3 assumes that a desired order k of the macromodel is provided as input. If, instead of the order, an a-priori error criterion in the $\mathcal{H}_\infty$-norm is supplied, the condition in step 9 is replaced by an inequality comparing the $\mathcal{H}_\infty$ error of the current model to the desired accuracy (note that the $\mathcal{H}_\infty$ error is already available as the largest error in the singular values computed at step 6).

Algorithm 4 Real adaptive approach with random sampling directions.

1: Generate a set of N_p indices linearly distributed between 1 and N_s.
2: Set $m \leftarrow N_p$.
3: Construct the order m system by building the state-space matrices as described in section 5.1.3.
4: **for** $i \leftarrow 1 \ldots N_s$ **do**
5: Evaluate the transfer function at sample point $j\omega_i$.
6: Compute the singular values of the difference between the data matrix at point ω_i and the transfer function evaluated at $j\omega_i$ and keep only the largest.
7: **end for**
8: Build a vector of indices which have already been used.
9: **while** m less than k, the desired order of the reduced system **do**
10: Select the two measurements which have not been used and give the largest errors in singular values over all N_s measurements and update $m \leftarrow m + 2$, as well as the vector of already used indices.
11: Construct the order m system as described in section 5.1.3, by preserving the system built at the previous step and appending the last two columns and rows, obtained from combining the two new measurements with all the previous ones.
12: Repeat steps 4 - 7.
13: **end while**

Algorithm 4 assumes that a desired order k of the macromodel is provided as input. If, instead of the order, an a-priori error criterion in the $\mathcal{H}_\infty$-norm is supplied, the condition in step 9 is replaced by an inequality comparing the $\mathcal{H}_\infty$ error of the current model to the desired accuracy (note that the $\mathcal{H}_\infty$ error is already available as the largest error in the singular values computed at step 6).

Algorithm 5 Complex adaptive approach with sampling directions chosen as the singular vectors associated with the largest singular value of the error matrix.

1: Generate a set of N_p indices linearly distributed between 1 and N_s.
2: Set $m \leftarrow N_p$.
3: Construct the order m system by building the state-space matrices as described in section 5.1.1, (5.11) and (5.12).
4: **for** $i \leftarrow 1 \ldots N_s$ **do**
5: Evaluate the transfer function at sample point $j\omega_i$.
6: Compute the singular values of the difference between the data matrix at point ω_i and the transfer function evaluated at $j\omega_i$ and keep only the largest.
7: **end for**
8: Build a vector of indices which have already been used.
9: **while** m less than k, the desired order of the reduced system **do**
10: Select the measurement which has not been used and gives the largest error in singular values over all N_s measurements and update $m \leftarrow m + 1$, as well as the vector of already used indices.
11: Construct the order m system as described in section 5.1.1, by preserving the system built at the previous step and appending the last column and row, obtained from combining the new measurement with all the previous ones. The sampling directions for the new measurement are taken as the singular vectors associated with the largest singular value of the error matrix.
12: Repeat steps 4 - 7.
13: **end while**

The algorithm presented above assumes that a desired order k of the macromodel is provided as input. If, instead of the order, an a-priori error criterion in the $\mathcal{H}_\infty$-norm is supplied, the condition in step 9 is replaced by an inequality comparing the $\mathcal{H}_\infty$ error of the current model to the desired accuracy (note that the $\mathcal{H}_\infty$ error is already available as the largest error in the singular values computed at step 6).

Algorithm 6 Real adaptive approach with sampling directions chosen as the singular vectors associated with the largest singular value of the error matrix.

1: Generate a set of N_p indices linearly distributed between 1 and N_s.
2: Set $m \leftarrow N_p$.
3: Construct the order m system by building the state-space matrices as described in section 5.1.3, (5.22) and (5.23).
4: **for** $i \leftarrow 1 \ldots N_s$ **do**
5: Evaluate the transfer function at the sample point $j\omega_i$.
6: Compute the singular values of the difference between the data matrix at point ω_i and the transfer function evaluated at $j\omega_i$ and keep only the largest.
7: **end for**
8: Build a vector of indices which have already been used.
9: **while** m less than k, the desired order of the reduced system **do**
10: Select the two measurements which have not been used and give the largest errors in singular values over all N_s measurements and update $m \leftarrow m + 2$, as well as the vector of already used indices.
11: Construct the order m system as described in section 5.1.3, by preserving the system built at the previous step and appending the last two columns and rows, obtained from combining the new measurement with all the previous ones. The sampling directions for the new measurements are taken as the singular vectors and the complex conjugates of the singular vectors, associated with the largest singular value of the two error matrices which gave the largest deviation.
12: Repeat steps 4 - 7.
13: **end while**

Algorithm 6 assumes that a desired order k of the macromodel is provided as input. If, instead of the order, an a-priori error criterion in the $\mathcal{H}_\infty$-norm is supplied, the condition in step 9 is replaced by an inequality comparing the $\mathcal{H}_\infty$ error of the current model to the desired accuracy (note that the $\mathcal{H}_\infty$ error is already available as the largest error in the singular values computed at step 6).

Algorithm 7 Complex adaptive approach with sampling directions chosen as the sum of the singular vectors of the error matrix.

1: Generate a set of N_p indices linearly distributed between 1 and N_s.
2: Set $m \leftarrow N_p$.
3: Construct the order m system by building the state-space matrices as described in section 5.1.1, (5.11) and (5.12).
4: **for** $i \leftarrow 1 \ldots N_s$ **do**
5: Evaluate the transfer function at sample point $j\omega_i$.
6: Compute the singular values of the difference between the data matrix at point ω_i and the transfer function evaluated at $j\omega_i$ and keep only the largest.
7: **end for**
8: Build a vector of indices which have already been used.
9: **while** m less than k, the desired order of the reduced system **do**
10: Select the measurement which has not been used and gives the largest error in singular values over all N_s measurements and update $m \leftarrow m + 1$, as well as the vector of already used indices.
11: Construct the order m system as described in section 5.1.1, by preserving the system built at the previous step and appending the last column and row, obtained from combining the new measurement with all the previous ones. The sampling directions for the new measurement are taken as the sum of the singular vectors of the error matrix which leads to the largest error.
12: Repeat steps 4 - 7.
13: **end while**

Algorithm 7 assumes that a desired order k of the macromodel is provided as input. If, instead of the order, an a-priori error criterion in the $\mathcal{H}_\infty$-norm is supplied, the condition in step 9 is replaced by an inequality comparing the $\mathcal{H}_\infty$ error of the current model to the desired accuracy (note that the $\mathcal{H}_\infty$ error is already available as the largest error in the singular values computed at step 6).

Algorithm 8 Real adaptive approach with sampling directions chosen as the sum of the singular vectors of the error matrix.

1: Generate a set of N_p indices linearly distributed between 1 and N_s.
2: Set $m \leftarrow N_p$.
3: Construct the order m system by building the corresponding state-space matrices as described in section 5.1.3, (5.22) and (5.23).
4: **for** $i \leftarrow 1 \ldots N_s$ **do**
5: Evaluate the transfer function at sample point $j\omega_i$.
6: Compute the singular values of the difference between the data matrix at point ω_i and the transfer function evaluated at $j\omega_i$ and keep only the largest.
7: **end for**
8: Build a vector of indices which have already been used.
9: **while** m less than k, the desired order of the model **do**
10: Select the two measurements which have not been used and give the largest error in singular values of the error matrices over all N_s measurements and update $m \leftarrow m + 2$, as well as the vector of already used indices.
11: Construct the order m system as described in section 5.1.3, by preserving the system built at the previous step and appending the last two columns and rows, obtained from combining the new measurements with all the previous ones. The sampling directions for the new measurements are taken as the sum of the singular vectors, together with their complex conjugates, of the two error matrices which gave the largest deviation.
12: Repeat steps 4 - 7.
13: **end while**

The algorithm presented above assumes that a desired order k of the macromodel is provided as input. If, instead of the order, an a-priori error criterion in the $\mathcal{H}_\infty$-norm is supplied, the condition in step 9 is replaced by an inequality comparing the $\mathcal{H}_\infty$ error of the current model to the desired accuracy (note that the $\mathcal{H}_\infty$ error is already available as the largest error in the singular values computed at step 6).

Algorithm 9 Complex adaptive approach with block processing and random sampling directions.

1: Generate a set of N_p indices linearly distributed between 1 and N_s.
2: Set $m \leftarrow N_p$.
3: Construct the order m system by building the corresponding state-space matrices as described in section 5.1.1.
4: **for** $i \leftarrow 1 \ldots N_s$ **do**
5: Evaluate the transfer function at sample point $j\omega_i$.
6: Compute the singular values of the difference between the data matrix at point ω_i and the transfer function evaluated at $j\omega_i$.
7: **end for**
8: **for** $i \leftarrow 1 \ldots N_p$ **do**
9: Order the i-th singular values of the error matrices in descending order and keep the sorted indices.
10: **end for**
11: Build a vector of indices which have already been used.
12: **while** m less than k, the desired order of the model **do**
13: **for** $i \leftarrow 1 \ldots N_p$ **do**
14: Select the first measurement indicated by the sorted indices which has not been seen and update $m \leftarrow m + 1$, as well as the vector of already used indices.
15: **end for**
16: Construct the order m system as described in section 5.1.1, by preserving the system built at the previous step and appending the last N_p columns or rows to the old matrices, obtained from combining the measurements indicated by the new indices with all the previous ones.
17: Repeat steps 4 - 10.
18: **end while**

The order of the models which are constructed with this algorithm is a multiple of the number of ports. However, it can be easily adapted to include extra h steps, where $k = N_p \cdot N + h$, in which one measurement is chosen for each of the first h largest singular values.

The algorithm presented above assumes that a desired order k of the macromodel is provided as input. If, instead of the order, an a-priori error criterion in the $\mathcal{H}_\infty$-norm is supplied, the condition in step 12 is replaced by an inequality comparing the $\mathcal{H}_\infty$ error of the current model to the desired accuracy (note that the $\mathcal{H}_\infty$ error is already available as the largest error in the singular values computed at step 6).

Algorithm 10 Complex adaptive approach with block processing and collapsing at the last step, using random sampling directions.

1: Generate a set of N_p indices linearly distributed between 1 and N_s.
2: Set $m \leftarrow N_p$.
3: Construct the order m system with state-space matrices as described in section 5.1.1.
4: **for** $i \leftarrow 1 \ldots N_s$ **do**
5: Evaluate the transfer function at sample point $j\omega_i$.
6: Compute the singular values of the difference between the data matrix at point ω_i and the transfer function evaluated at $j\omega_i$.
7: **end for**
8: **for** $i \leftarrow 1 \ldots N_p$ **do**
9: Order the i-th singular values of the error matrices in descending order and keep the sorted indices.
10: **end for**
11: Build a vector of indices which have already been used.
12: **while** m less than $k + N_p$ **do**
13: **for** $i \leftarrow 1 \ldots N_p$ **do**
14: Select the first measurement indicated by the sorted indices which has not been seen and update $m \leftarrow m + 1$, as well as the vector of already used indices.
15: **end for**
16: Construct the order m system as described in section 5.1.1, by preserving the system built at the previous steps and appending the last N_p columns or rows to the old matrices, obtained from combining the measurements indicated by the new indices with all the previous ones.
17: Repeat steps 4 - 7.
18: **end while**
19: Determine the projectors $\mathbf{X}$ and $\mathbf{Y}$ from (5.14).
20: Compute the corresponding state-space matrices using the formulas in (5.15).

The order of the models constructed with algorithm 10 is a multiple of the number of ports. However, it can be easily adapted to include extra h steps, where $k = N_p \cdot N + h$, in which one measurement is chosen for each of the first h largest singular values. It assumes that a desired order k of the model is provided as input. If an a-priori error criterion in the $\mathcal{H}_\infty$-norm is supplied, the condition in step 12 is replaced by an inequality comparing the $\mathcal{H}_\infty$ error of the current model to the desired accuracy (the $\mathcal{H}_\infty$ error is already available as the largest error in the singular values computed at step 6).

Algorithm 11 Complex adaptive approach with block processing and collapsing at the last step, having the sampling directions as the singular vectors associated with the largest singular value of the error matrix.

1: Generate a set of N_p indices linearly distributed between 1 and N_s.
2: Set $m \leftarrow N_p$.
3: Construct the order m system with state-space matrices as described in section 5.1.1.
4: **for** $i \leftarrow 1 \ldots N_s$ **do**
5: Evaluate the transfer function at sample point $j\omega_i$.
6: Compute the singular values of the difference between the data matrix at point ω_i and the transfer function evaluated at $j\omega_i$.
7: **end for**
8: **for** $i \leftarrow 1 \ldots N_p$ **do**
9: Order the i-th singular values of the error matrices in descending order and keep the sorted indices.
10: **end for**
11: Build a vector of indices which have already been used.
12: **while** m less than $k + N_p$ **do**
13: **for** $i \leftarrow 1 \ldots N_p$ **do**
14: Select the first measurement indicated by the sorted indices which has not been seen and update $m \leftarrow m + 1$, as well as the vector of already used indices.
15: **end for**
16: Construct the order m system as described in section 5.1.1, by preserving the system built at the previous steps and appending columns or rows to the old matrices, obtained from combining the measurements indicated by the new indices with all the previous ones. The sampling directions of the new measurements are chosen as the singular vectors associated with the corresponding singular value which gave the largest error.
17: Repeat steps 4 - 7.
18: **end while**
19: Determine the projectors $\mathbf{X}$ and $\mathbf{Y}$ from (5.14).
20: Compute the corresponding state-space matrices using the formulas in (5.15).

The order of the models constructed with algorithm 11 is a multiple of the number of ports. However, it can be easily adapted to include extra h steps, where $k = N_p \cdot N + h$, in which one measurement is chosen for each of the first h largest singular values. It assumes that a desired order k of the model is provided as input. If an a-priori error criterion in the $\mathcal{H}_\infty$-norm is supplied, the condition in step 12 is replaced by an inequality comparing the $\mathcal{H}_\infty$ error of the current model to the desired accuracy

(the $\mathcal{H}_\infty$ error is already available as the largest error in the singular values computed at step 6).

Algorithm 12 Complex adaptive approach with random sampling directions, reusing measurements.

1: Generate a set of N_p indices linearly distributed between 1 and N_s.
2: Set $m \leftarrow N_p$.
3: Construct the order m system by building the associated state-space matrices as described in section 5.1.1.
4: **for** $i \leftarrow 1 \ldots N_s$ **do**
5: Evaluate the transfer function at the sample point $j\omega_i$.
6: Compute the singular values of the difference between the data matrix at point ω_i and the transfer function evaluated at $j\omega_i$ and keep only the largest.
7: **end for**
8: **while** m less than k, the desired order of the reduced system **do**
9: Select the measurement which gives the largest error in singular values over all N_s measurements and update $m \leftarrow m + 1$.
10: Construct the order m system as described in section 5.1.1, by preserving the system built at the previous step and appending the last column and row, obtained from combining the new measurement with all the previous ones.
11: Repeat steps 4 - 7.
12: **end while**

The algorithm presented above assumes that a desired order k of the macromodel is provided as input. If, instead of the order, an a-priori error criterion in the $\mathcal{H}_\infty$-norm is supplied, the condition in step 8 is replaced by an inequality comparing the $\mathcal{H}_\infty$ error of the current model to the desired accuracy (note that the $\mathcal{H}_\infty$ error is already available as the largest error in the singular values computed at step 6).

Algorithm 13 Real adaptive approach with random sampling directions and reusing measurements.

1: Generate a set of N_p indices linearly distributed between 1 and N_s.
2: Set $m \leftarrow N_p$.
3: Construct the order m system by building the corresponding state-space matrices as described in section 5.1.3.
4: **for** $i \leftarrow 1 \ldots N_s$ **do**
5: Evaluate the transfer function at sample point $j\omega_i$.
6: Compute the singular values of the difference between the data matrix at point ω_i and the transfer function evaluated at $j\omega_i$ and keep only the largest one of the N_p ones.
7: **end for**
8: Build two vectors of indices which have already been used as left and right tangential data, respectively.
9: **while** m less than k, the desired order of the reduced system **do**
10: Select the two measurements which give the largest error in singular values over all N_s measurements and update $m \leftarrow m + 2$.
11: Construct the order m system as described in section 5.1.3, by preserving the system built at the previous step and appending the last two columns and rows, obtained from combining the new measurements with all the previous ones. If the new measurement has already been used, check whether it was used as left or right data and use it in the same manner. Last, update the vectors of indices which were already used.
12: Repeat steps 4 - 7.
13: **end while**

Algorithm 13 assumes that a desired order k of the macromodel is provided as input. If, instead of the order, an a-priori error criterion in the $\mathcal{H}_\infty$-norm is supplied, the condition in step 9 is replaced by an inequality comparing the $\mathcal{H}_\infty$ error of the current model to the desired accuracy (note that the $\mathcal{H}_\infty$ error is already available as the largest error in the singular values computed at step 3).

Algorithm 14 Complex adaptive approach with sampling directions chosen as the singular vectors associated with the largest singular value of the error matrix, reusing measurements.

1: Generate a set of N_p indices linearly distributed between 1 and N_s.
2: Set $m \leftarrow N_p$.
3: Construct the order m system by building the corresponding state-space matrices as described in section 5.1.1, (5.11) and (5.12).
4: **for** $i \leftarrow 1 \ldots N_s$ **do**
5: Evaluate the transfer function at sample point $j\omega_i$.
6: Compute the singular values of the difference between the data matrix at point ω_i and the transfer function evaluated at $j\omega_i$ and keep only the largest.
7: **end for**
8: **while** m less than k, the desired order of the reduced system **do**
9: Select the measurement which gives the largest error in singular values over all N_s measurements and update $m \leftarrow m + 1$.
10: Construct the order m system as described in section 5.1.1, by preserving the system built at the previous step and appending the last column and row, obtained from combining the new measurement with all the previous ones. The sampling directions for the new measurement are taken as the singular vectors associated with the largest singular value of the error matrix.
11: Repeat steps 4 - 7.
12: **end while**

Algorithm 14 assumes that a desired order k of the macromodel is provided as input. If, instead of the order, an a-priori error criterion in the $\mathcal{H}_\infty$-norm is supplied, the condition in step 8 is replaced by an inequality comparing the $\mathcal{H}_\infty$ error of the current model to the desired accuracy (note that the $\mathcal{H}_\infty$ error is already available as the largest error in the singular values computed at step 6).

Algorithm 15 Real adaptive approach with sampling directions chosen as the singular vectors associated with the two largest singular values of the error matrices, reusing measurements.

1: Generate a set of N_p indices linearly distributed between 1 and N_s.
2: Set $m \leftarrow N_p$.
3: Construct the order m system by building the corresponding state-space matrices as described in section 5.1.3, (5.22) and (5.23).
4: **for** $i \leftarrow 1 \ldots N_s$ **do**
5: Evaluate the transfer function at sample point $j\omega_i$.
6: Compute the singular values of the difference between the data matrix at point ω_i and the transfer function evaluated at $j\omega_i$ and keep only the largest.
7: **end for**
8: Build two vectors of indices which have already been used as left and right tangential data, respectively.
9: **while** m less than k, the desired order of the reduced system **do**
10: Select the two measurements which give the largest error in singular values over all N_s measurements and update $m \leftarrow m + 2$.
11: Construct the order m system as described in section 5.1.3, by preserving the system built at the previous step and appending the last two columns and rows, obtained from combining the new measurements with all the previous ones. If the new measurement has already been used, check whether it was used as left or right data and use it in the same manner. The sampling directions for the new measurements are taken as the singular vectors associated with the largest singular value of the two error matrices which gave the largest deviation. Last, update the vectors of indices which were already used.
12: Repeat steps 4 - 7.
13: **end while**

The algorithm presented above assumes that a desired order k of the macromodel is provided as input. If, instead of the order, an a-priori error criterion in the $\mathcal{H}_\infty$-norm is supplied, the condition in step 9 is replaced by an inequality comparing the $\mathcal{H}_\infty$ error of the current model to the desired accuracy (note that the $\mathcal{H}_\infty$ error is already available as the largest error in the singular values computed at step 6).

Algorithm 16 Complex adaptive approach with sampling directions chosen as the sum of the singular vectors of the error matrix, reusing measurements.

1: Generate a set of N_p indices linearly distributed between 1 and N_s.
2: Set $m \leftarrow N_p$.
3: Construct the order m system by building the corresponding state-space matrices as described in section 5.1.1, (5.11) and (5.12).
4: **for** $i \leftarrow 1 \ldots N_s$ **do**
5: Evaluate the transfer function at sample point $j\omega_i$.
6: Compute the singular values of the error matrix at point ω_i computed as the difference between the data matrix at point ω_i and the transfer function evaluated at $j\omega_i$ and keep only the largest.
7: **end for**
8: **while** m less than k, the desired order of the reduced system **do**
9: Select the measurement which gives the largest error in singular values over all N_s measurements and update $m \leftarrow m + 1$.
10: Construct the order m system as described in section 5.1.1, by preserving the system built at the previous step and appending the last column and row, obtained from combining the new measurement with all the previous ones. The sampling directions for the new measurement are taken as the sum of the singular vectors of the error matrix at this measurement.
11: Repeat steps 4 - 7.
12: **end while**

Algorithm 16 assumes that a desired order k of the macromodel is provided as input. If, instead of the order, an a-priori error criterion in the $\mathcal{H}_\infty$-norm is supplied, the condition in step 8 is replaced by an inequality comparing the $\mathcal{H}_\infty$ error of the current model to the desired accuracy (note that the $\mathcal{H}_\infty$ error is already available as the largest error in the singular values computed at step 6).

Algorithm 17 Real adaptive approach with sampling directions chosen as the sum of the singular vectors of the error matrix, reusing measurements.

1: Generate a set of N_p indices linearly distributed between 1 and N_s.
2: Set $m \leftarrow N_p$.
3: Construct the order m system by building the corresponding state-space matrices as described in section 5.1.3, (5.22) and (5.23).
4: **for** $i \leftarrow 1 \ldots N_s$ **do**
5: Evaluate the transfer function at sample point $j\omega_i$.
6: Compute the singular values of the difference between the data matrix at point ω_i and the transfer function evaluated at $j\omega_i$ and keep only the largest.
7: **end for**
8: Build two vectors of indices which have already been used as left and right tangential data, respectively.
9: **while** m less than k, the desired order of the reduced system **do**
10: Select the two measurements which give the largest error in singular values over all N_s measurements and update $m \leftarrow m + 2$.
11: Construct the order m system as described in section 5.1.3, by preserving the system built at the previous step and appending the last two columns and rows, obtained from combining the new measurements with all the previous ones. If the new measurement has already been used, check whether it was used as left or right data and use it in the same manner. The sampling directions for the new measurements are taken as the sum of the singular vectors of the two error matrices which gave the largest deviation. Last, update the vectors of indices which were already used.
12: Repeat steps 4 - 7.
13: **end while**

The algorithm presented above assumes that a desired order k of the macromodel is provided as input. If, instead of the order, an a-priori error criterion in the $\mathcal{H}_\infty$-norm is supplied, the condition in step 9 is replaced by an inequality comparing the $\mathcal{H}_\infty$ error of the current model to the desired accuracy (note that the $\mathcal{H}_\infty$ error is already available as the largest error in the singular values computed at step 6).

Algorithm 18 Complex adaptive approach with block processing and random sampling directions, reusing measurements.

1: Generate a set of N_p indices linearly distributed between 1 and N_s.
2: Set $m \leftarrow N_p$.
3: Construct the order m system by building the corresponding state-space matrices as described in section 5.1.1.
4: **for** $i \leftarrow 1 \ldots N_s$ **do**
5: Evaluate the transfer function at sample point $j\omega_i$.
6: Compute the singular values of the difference between the data matrix at point ω_i and the transfer function evaluated at $j\omega_i$.
7: **end for**
8: **for** $i \leftarrow 1 \ldots N_p$ **do**
9: Find the maximum i-th singular value of all error matrices and the corresponding index.
10: **end for**
11: **while** m less than k, the desired order of the model **do**
12: Update $m \leftarrow m + N_p$.
13: Construct the order m system as described in section 5.1.1, by preserving the system built at the previous step and appending the last N_p columns or rows to the old matrices, obtained from combining the measurements indicated by the new indices with all the previous ones. For each new measurement, the sampling directions are randomly generated complex vectors.
14: Repeat steps 4 - 10.
15: **end while**

The order of the models which are constructed with this algorithm is a multiple of the number of ports. However, it can be easily adapted to include extra h steps, where $k = N_p \cdot N + h$, in which one measurement is chosen for each of the first h largest singular values.

Algorithm 18 assumes that a desired order k of the macromodel is provided as input. If, instead of the order, an a-priori error criterion in the $\mathcal{H}_\infty$-norm is supplied, the condition in step 11 is replaced by an inequality comparing the $\mathcal{H}_\infty$ error of the current model to the desired accuracy (note that the $\mathcal{H}_\infty$ error is already available as the largest error in the singular values computed at step 6).

Algorithm 19 Complex adaptive approach with block processing and collapsing at the last step, using random sampling directions and reusing measurements.

1: Generate a set of N_p indices linearly distributed between 1 and N_s.
2: Set $m \leftarrow N_p$.
3: Construct the order m system by building the corresponding state-space matrices as described in section 5.1.1.
4: **for** $i \leftarrow 1 \ldots N_s$ **do**
5: Evaluate the transfer function at sample point $j\omega_i$.
6: Compute the singular values of the difference between the data matrix at point ω_i and the transfer function evaluated at $j\omega_i$.
7: **end for**
8: **for** $i \leftarrow 1 \ldots N_p$ **do**
9: Find the maximum i-th singular value of all error matrices and the corresponding index.
10: **end for**
11: **while** m less than $k + N_p$ **do**
12: Update $m \leftarrow m + N_p$.
13: Construct the order m system as described in section 5.1.1, by preserving the system built at the previous step and appending the last N_p columns or rows to the old matrices, obtained from combining the measurements indicated by the new indices with all the previous ones. For each new measurement, the sampling directions are randomly generated complex vectors.
14: Repeat steps 4 - 10.
15: **end while**
16: Determine the projectors $\mathbf{X}$ and $\mathbf{Y}$ from (5.14).
17: Compute the corresponding state-space matrices using the formulas in (5.15).

The order of the models which are constructed with this algorithm is a multiple of the number of ports. However, it can be easily adapted to include extra h steps, where $k = N_p \cdot N + h$, in which one measurement is chosen for each of the first h largest singular values.

Algorithm 19 assumes that a desired order k of the macromodel is provided as input. If, instead of the order, an a-priori error criterion in the $\mathcal{H}_\infty$-norm is supplied, the condition in step 11 is replaced by an inequality comparing the $\mathcal{H}_\infty$ error of the current model to the desired accuracy (note that the $\mathcal{H}_\infty$ error is already available as the largest error in the singular values computed at step 6).

Algorithm 20 Complex adaptive approach with block processing and collapsing at the last step, having the sampling directions as the singular vectors associated with the largest singular value of the error matrix, reusing measurements.

1: Generate a set of N_p indices linearly distributed between 1 and N_s.
2: Set $m \leftarrow N_p$.
3: Construct the order m system by building the corresponding state-space matrices as described in section 5.1.1.
4: **for** $i \leftarrow 1 \ldots N_s$ **do**
5: Evaluate the transfer function at sample point $j\omega_i$.
6: Compute the singular values of the difference between the data matrix at point ω_i and the transfer function evaluated at $j\omega_i$.
7: **end for**
8: **for** $i \leftarrow 1 \ldots N_p$ **do**
9: Find the maximum i-th singular value of all error matrices and the corresponding index.
10: **end for**
11: **while** m less than $k + N_p$ **do**
12: Update $m \leftarrow m + N_p$.
13: Construct the order m system as described in section 5.1.1, by preserving the system built at the previous step by appending the last N_p columns or rows to the old matrices, obtained from combining the measurements indicated by the new indices with all the previous ones. The sampling directions of the new measurements are chosen as the singular vectors associated with the corresponding singular value which gave the largest error.
14: Repeat steps 4 - 10.
15: **end while**
16: Determine the projectors $\mathbf{X}$ and $\mathbf{Y}$ from (5.14).
17: Compute the corresponding state-space matrices using the formulas in (5.15).

The order of the models constructed with algorithm 20 is a multiple of the number of ports. However, it can be easily adapted to include extra h steps, where $k = N_p \cdot N + h$, in which one measurement is chosen for each of the first h largest singular values. It assumes that a desired order k of the model is provided as input. If an a-priori error criterion in the $\mathcal{H}_\infty$-norm is supplied, the condition in step 11 is replaced by an inequality comparing the $\mathcal{H}_\infty$ error of the current model to the desired accuracy (the $\mathcal{H}_\infty$ error is already available as the largest error in the singular values computed at step 6).

Algorithm 21 Recursive construction of interpolants in an adaptive fashion using the complex approach.

for $i \leftarrow 1 \ldots N_s$ **do**
  Generate random sampling directions $\mathbf{r}_{o,i}$, set $\ell_{o,i} = \mathbf{r}^*_{o,i}$, compute $\mathbf{w}_{o,i} = \mathbf{S}_i \mathbf{r}_{o,i}$ and $\mathbf{v}_{o,i} = \ell_{o,i} \bar{\mathbf{S}}_i$.
end for
Construct the order $m \leftarrow 1$ interpolant from Eq. (5.41)-(5.44).
for $i \leftarrow 1 \ldots N_s$ **do**
  Evaluate the current generating system and its inverse at the i-th frequency sample and its conjugate, to compute the errors $\ell_{e,i}$, $\mathbf{v}_{e,i}$, $\mathbf{w}_{e,i}$ and $\mathbf{r}_{e,i}$ as given by Eq. (5.45)-(5.46).
end for
Select the measurement with the largest magnitude of the error vectors.
Compute $\mathbb{L}_2$ from 5.47
while $\mathbb{L}_2 \geq tol$, an a-priori tolerance **do**
  Construct the order $m+1$ interpolant using Eq. (4.59)-(4.62) and (5.47)-(5.50).
  Update $m \leftarrow m + 1$.
  for $i \leftarrow 1 \ldots N_s$ **do**
    Evaluate the current generating system and its inverse at the i-th frequency sample and its conjugate, to compute the errors $\ell_{e,i}$, $\mathbf{v}_{e,i}$, $\mathbf{w}_{e,i}$ and $\mathbf{r}_{e,i}$ as given by Eq. (5.45)-(5.46).
  end for
  Select the measurement with the largest magnitude of the error vectors.
  Compute $\mathbb{L}_2$ from 5.47
end while
Compute $\sigma\mathbb{L}_2$ from 5.48
if $\sigma\mathbb{L}_2 \geq tol$, an a-priori tolerance **then**
  for $i \leftarrow 1 \ldots N_s$ **do**
    Evaluate the current generating system and its inverse at the i-th frequency sample and its conjugate, to compute the errors $\ell_{e,i}$, $\mathbf{v}_{e,i}$, $\mathbf{w}_{e,i}$ and $\mathbf{r}_{e,i}$ as given by Eq. (5.45)-(5.46).
  end for
  Sort the $\mathbf{v}_{e,i}$ vectors in descending order according to their norm and keep the sorted indices.
  Construct $\mathbb{L}_2$, $\sigma\mathbb{L}_2$, $\mathbf{V}_2$ and $\mathbf{W}_2$ from the new indices.
  Construct the order $m + N_p$ interpolant using Eq. (4.59)-(4.62).
else
  Return the order m system: $\mathbf{E}$ as $-\mathbb{L}_1$, $\mathbf{A}$ as $-\sigma\mathbb{L}_1$, $\mathbf{B}$ as $\mathbf{V}_1$, $\mathbf{C}$ as $\mathbf{W}_1$ and $\mathbf{D}$ as $\mathbf{0}$.
end if

Algorithm 22 Recursive construction of interpolants in an adaptive fashion using the real approach.

for $i \leftarrow 1 \ldots \frac{N_s}{2}$ **do**
  Generate random sampling directions $\ell_{o,2i-1}$, $\mathbf{r}_{o,2i-1}$, set $\ell_{o,2i} = \bar{\ell}_{o,2i-1}$, $\mathbf{r}_{o,2i} = \bar{\mathbf{r}}_{o,2i-1}$, compute $\mathbf{w}_{o,2i-1} = \mathbf{S}_i \mathbf{r}_{o,2i-1}$, $\mathbf{v}_{o,2i-1} = \ell_{o,2i-1} \mathbf{S}_{\frac{N_s}{2}+i}$ and set $\mathbf{w}_{o,2i} = \bar{\mathbf{w}}_{o,2i-1}$ and $\mathbf{v}_{o,2i} = \bar{\mathbf{v}}_{o,2i-1}$.
end for
Construct the order $m \leftarrow 2$ interpolant from Eq. (5.51)-(5.54).
for $i \leftarrow 1, 3, \ldots N_s - 1$ **do**
  Evaluate the current generating system at $j\omega_{\frac{i+1}{2}}$ to compute the errors $\mathbf{w}_{e,i}$ and $\mathbf{r}_{e,i}$; also evaluate the inverse of the current generating system at $j\omega_{\frac{i+1+N_s}{2}}$ to compute the errors $\ell_{e,i}$, $\mathbf{v}_{e,i}$, as given by Eq. (5.45)-(5.46).
end for
Select the measurement with the largest magnitude of the error vectors $\mathbf{w}_e$ as well as the measurement with the largest magnitude of the error vectors $\mathbf{v}_e$.
Compute $\mathbb{L}_2$ from 5.47
while norm($\mathbb{L}_2$) $\geq$ *tol*, an a-priori tolerance **do**
  Construct the order $m + 2$ interpolant using Eq. (4.59)-(4.62).
  Update $m \leftarrow m + 2$.
  for $i \leftarrow 1, 3, \ldots N_s - 1$ **do**
    Evaluate the current generating system at $j\omega_{\frac{i+1}{2}}$ to compute the errors $\mathbf{w}_{e,i}$ and $\mathbf{r}_{e,i}$; also evaluate the inverse of the current generating system at $j\omega_{\frac{i+1+N_s}{2}}$ to compute the errors $\ell_{e,i}$, $\mathbf{v}_{e,i}$, as given by Eq. (5.45)-(5.46).
  end for
  Select the measurement with the largest magnitude of the error vectors $\mathbf{w}_e$ as well as the measurement with the largest magnitude of the error vectors $\mathbf{v}_e$.
end while
Compute $\sigma\mathbb{L}_2$.
if norm($\sigma\mathbb{L}_2$) $\geq$ *tol*, an a-priori tolerance **then**
  for $i \leftarrow 1 \ldots N_s$ **do**
    Evaluate the current generating system and its inverse at the i-th frequency sample and its conjugate, to compute the errors $\ell_{e,i}$, $\mathbf{v}_{e,i}$, $\mathbf{w}_{e,i}$ and $\mathbf{r}_{e,i}$ as given by Eq. (5.45)-(5.46).
  end for
  Sort the $\mathbf{v}_{e,i}$ vectors in descending order according to their norm and keep the sorted indices.
  Construct $\mathbb{L}_2$, $\sigma\mathbb{L}_2$, $\mathbf{V}_2$ and $\mathbf{W}_2$ from the new indices.
  Construct the order $m + N_p$ interpolant using Eq. (4.59)-(4.62).
else
  Return the order m system: $\mathbf{E}$ as $-\mathbb{L}_1$, $\mathbf{A}$ as $-\sigma\mathbb{L}_1$, $\mathbf{B}$ as $\mathbf{V}_1$, $\mathbf{C}$ as $\mathbf{W}_1$ and $\mathbf{D}$ as $\mathbf{0}$.
end if

Algorithm 23 Recursive construction of interpolants in an adaptive fashion using the complex approach, allowing for measurements to be used more than once.

for $i \leftarrow 1 \ldots N_s$ **do**
  Generate random sampling directions $\mathbf{r}_{o,i}$, set $\ell_{o,i} = \mathbf{r}^*_{o,i}$, compute $\mathbf{w}_{o,i} = \mathbf{S}_i \mathbf{r}_{o,i}$ and $\mathbf{v}_{o,i} = \ell_{o,i} \bar{\mathbf{S}}_i$.
end for
Construct the order $m \leftarrow 1$ interpolant from Eq. (5.41)-(5.44) and substitute the previous $\ell_{o,k}$ and $\mathbf{r}_{o,k}$ vectors with new ones, still randomly generated. The $\mathbf{v}_{o,k}$ and $\mathbf{w}_{o,k}$ vectors also need to updated.
for $i \leftarrow 1 \ldots N_s$ **do**
  Evaluate the current generating system and its inverse at the i-th frequency sample and its conjugate, to compute the errors $\ell_{e,i}$, $\mathbf{v}_{e,i}$, $\mathbf{w}_{e,i}$ and $\mathbf{r}_{e,i}$ as given by Eq. (5.45)-(5.46).
end for
Select the measurement with the largest magnitude of the error vectors and substitute the previous $\ell_{o,k}$ and $\mathbf{r}_{o,k}$ vectors with new ones, still randomly generated. The $\mathbf{v}_{o,k}$ and $\mathbf{w}_{o,k}$ vectors also need to updated.
Compute $\mathbb{L}_2$ from 5.47
while $\mathbb{L}_2 \geq tol$, an a-priori tolerance **do**
  Construct the order $m+1$ interpolant using Eq. (4.59)-(4.62) and (5.47)-(5.50).
  Update $m \leftarrow m + 1$.
  for $i \leftarrow 1 \ldots N_s$ **do**
    Evaluate the current generating system and its inverse at the i-th frequency sample and its conjugate, to compute the errors $\ell_{e,i}$, $\mathbf{v}_{e,i}$, $\mathbf{w}_{e,i}$ and $\mathbf{r}_{e,i}$ as given by Eq. (5.45)-(5.46).
  end for
  Select the measurement with the largest magnitude of the error vectors and substitute the previous $\ell_{o,k}$ and $\mathbf{r}_{o,k}$ vectors with new ones, still randomly generated. The $\mathbf{v}_{o,k}$ and $\mathbf{w}_{o,k}$ vectors also need to updated.
  Compute $\mathbb{L}_2$ from 5.47
end while
Compute $\sigma\mathbb{L}_2$ from 5.48
if $\sigma\mathbb{L}_2 \geq tol$, an a-priori tolerance **then**
  for $i \leftarrow 1 \ldots N_s$ **do**
    Evaluate the current generating system and its inverse at the i-th frequency sample and its conjugate, to compute the errors $\ell_{e,i}$, $\mathbf{v}_{e,i}$, $\mathbf{w}_{e,i}$ and $\mathbf{r}_{e,i}$ as given by Eq. (5.45)-(5.46).
  end for
  Sort the $\mathbf{v}_{e,i}$ vectors in descending order according to their norm and keep the sorted indices.
  Construct $\mathbb{L}_2$, $\sigma\mathbb{L}_2$, $\mathbf{V}_2$ and $\mathbf{W}_2$ from the new indices.
  Construct the order $m + N_p$ interpolant using Eq. (4.59)-(4.62).
else
  Return the order m system: $\mathbf{E}$ as $-\mathbb{L}_1$, $\mathbf{A}$ as $-\sigma\mathbb{L}_1$, $\mathbf{B}$ as $\mathbf{V}_1$, $\mathbf{C}$ as $\mathbf{W}_1$ and $\mathbf{D}$ as $\mathbf{0}$.
end if

Algorithm 24 Recursive construction of interpolants in an adaptive fashion using the real approach, allowing for measurements to be used more than once.

for $i \leftarrow 1 \ldots \frac{N_s}{2}$ **do**
  Generate random sampling directions $\ell_{o,2i-1}$, $\mathbf{r}_{o,2i-1}$, set $\ell_{o,2i} = \bar{\ell}_{o,2i-1}$, $\mathbf{r}_{o,2i} = \bar{\mathbf{r}}_{o,2i-1}$, compute $\mathbf{w}_{o,2i-1} = \mathbf{S}_i \mathbf{r}_{o,2i-1}$, $\mathbf{v}_{o,2i-1} = \ell_{o,2i-1} \mathbf{S}_{\frac{N_s}{2}+i}$, set $\mathbf{w}_{o,2i} = \bar{\mathbf{w}}_{o,2i-1}$ and $\mathbf{v}_{o,2i} = \bar{\mathbf{v}}_{o,2i-1}$.
end for
Construct the order $m \leftarrow 2$ system from Eq. (5.51)-(5.54) and substitute the previous $\ell_{o,1}$ and $\mathbf{r}_{o,1}$ vectors with new ones, still randomly chosen, and set $\ell_{o,2} = \bar{\ell}_{o,1}$ and $\mathbf{r}_{o,2} = \bar{\mathbf{r}}_{o,1}$. Update $\mathbf{v}_{o,1}$ and $\mathbf{w}_{o,1}$ and $\mathbf{v}_{o,2} = \bar{\mathbf{v}}_{o,1}$, $\mathbf{w}_{o,2} = \bar{\mathbf{w}}_{o,1}$.
for $i \leftarrow 1, 3, \ldots N_s - 1$ **do**
  Evaluate the current generating system at $j\omega_{\frac{i+1}{2}}$ to compute the errors $\mathbf{w}_{e,i}$ and $\mathbf{r}_{e,i}$ and the inverse of the current generating system at $j\omega_{\frac{i+1+N_s}{2}}$ to compute the errors $\ell_{e,i}$, $\mathbf{v}_{e,i}$, as given by Eq. (5.45)-(5.46).
end for
Select the measurement with the largest magnitude of the error vectors $\mathbf{w}_e$ and the one with the largest magnitude of the error vectors $\mathbf{v}_e$. Substitute the previous ℓ_o and $\mathbf{r}_o$ vectors with new ones, still randomly generated, update $\mathbf{v}_o$ and $\mathbf{w}_o$ and store their complex conjugate values at the next positions.
Compute $\mathbb{L}_2$ from 5.47
while norm($\mathbb{L}_2$) $\geq$ *tol*, an a-priori tolerance **do**
  Construct the order $m+2$ model from Eq. (4.59)-(4.62), update $m \leftarrow m+2$.
  for $i \leftarrow 1, 3, \ldots N_s - 1$ **do**
    Evaluate the current generating system at $j\omega_{\frac{i+1}{2}}$ to compute the errors $\mathbf{w}_{e,i}$ and $\mathbf{r}_{e,i}$ and the inverse of the current generating system at $j\omega_{\frac{i+1+N_s}{2}}$ to compute the errors $\ell_{e,i}$, $\mathbf{v}_{e,i}$, as given by Eq. (5.45)-(5.46).
  end for
  Select the measurement with the largest magnitude of the error vectors $\mathbf{w}_e$ and the one with the largest magnitude of the error vectors $\mathbf{v}_e$. Substitute the previous ℓ_o and $\mathbf{r}_o$ vectors with new ones, still randomly generated, update $\mathbf{v}_o$ and $\mathbf{w}_o$ and store their complex conjugate values at the next positions.
end while
Compute $\sigma\mathbb{L}_2$.
if norm($\sigma\mathbb{L}_2$) $\geq$ *tol*, an a-priori tolerance **then**
  for $i \leftarrow 1 \ldots N_s$ **do**
    Evaluate the current generating system and its inverse at the i-th frequency sample and its conjugate, to compute the errors $\ell_{e,i}$, $\mathbf{v}_{e,i}$, $\mathbf{w}_{e,i}$ and $\mathbf{r}_{e,i}$ as given by Eq. (5.45)-(5.46).
  end for
  Sort the $\mathbf{v}_{e,i}$ vectors in descending order according to their norm and keep the sorted indices.
  Construct $\mathbb{L}_2$, $\sigma\mathbb{L}_2$, $\mathbf{V}_2$ and $\mathbf{W}_2$ from the new indices.
  Construct the order $m + N_p$ interpolant using Eq. (4.59)-(4.62).
else
  Return the order m system: $\mathbf{E}$ as $-\mathbb{L}_1$, $\mathbf{A}$ as $-\sigma\mathbb{L}_1$, $\mathbf{B}$ as $\mathbf{V}_1$, $\mathbf{C}$ as $\mathbf{W}_1$ and $\mathbf{D}$ as $\mathbf{0}$.
end if

The new notations introduced are N_1, the number of iterations used to fit the sum of each column, and N_2, the number of iterations required to fit each column.

Algorithm 25 Vector Fitting

1: **for** $n \leftarrow 1, \ldots, N_s$ **do**
2: **for** $m \leftarrow 1, \ldots, N_p$ **do**
3: Compute $T_{m,n}$ as the sum of the measurement-matrix $\mathbf{S}_n$ along column m.
4: **end for**
5: **end for**
6: Generate a set of starting poles $\bar{a}_1, \ldots, \bar{a}_N$.
7: **for** $m \leftarrow 1, \ldots, N_p$ **do**
8: **for** $l \leftarrow 1 \ldots N_1$ **do**
9: Solve a least squares problem of the form $\mathbf{Ax} = \mathbf{b}$, where S_i in (3.5) and (3.7) is $T_{m,i}$, for $i = 1, \ldots, N_s$.
10: Compute the eigenvalues $\bar{z}_i$, $i = 1, \ldots, N$ of the matrix

$$diag(\bar{a}_1, \ldots, \bar{a}_N) - \begin{bmatrix} 1 & \ldots & 1 \end{bmatrix}^T \cdot \begin{bmatrix} \tilde{c}_1 & \ldots & \tilde{c}_N \end{bmatrix}$$

and update $\bar{a}_1 \leftarrow \bar{z}_1, \ldots, \bar{a}_N \leftarrow \bar{z}_N$.
11: **end for**
12: **for** $l \leftarrow 1 \ldots N_2$ **do**
13: Solve a least squares problem of the form $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{M}_i$ in (3.8) and (3.10) is $\mathbf{S}(:, m)_i$, for $i = 1, \ldots, N_s$.
14: Compute the eigenvalues $\bar{z}_i$, $i = 1, \ldots, N$ of the matrix

$$diag(\bar{a}_1, \ldots, \bar{a}_N) - \begin{bmatrix} 1 & \ldots & 1 \end{bmatrix}^T \cdot \begin{bmatrix} \tilde{c}_1 & \ldots & \tilde{c}_N \end{bmatrix}$$

and update $\bar{a}_1 \leftarrow \bar{z}_1, \ldots, \bar{a}_N \leftarrow \bar{z}_N$.
15: **end for**
16: Solve a least squares problem of the form $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{M}_i$ in (3.17) and (3.19) is $\mathbf{S}_i(:, m)$, for $i = 1, \ldots, N_s$.
17: **end for**

APPENDIX B

Further Numerical Examples

B.1 Noise-free SISO system with 4 poles

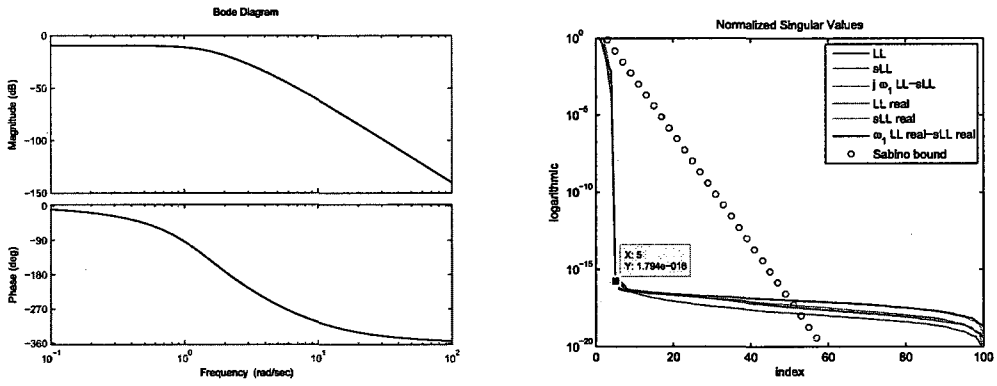
Consider the single-input single-output (SISO) bounded-real system given by the transfer function

$$H(s) = \frac{10}{[(s+1)^2+1](s+3)(s+5)}.$$

The system is of order 4 with poles at $-1 \pm j$, -3 and -5. We sample the transfer function at 100 linearly spaced frequency points between 10^{-1} rad/sec and 10^{1} rad/sec. We compare all the algorithms previously proposed in terms of computational time and normalized errors.

Fig.B.1(a) shows the Bode plot of the system, while Fig.B.1(b) shows the plot of the normalized singular values of the Loewner and shifted Loewner matrices constructed both using the complex as well as the real approach. The sampling left and right tangential directions were chosen as outlined in Eqs.(5.11), (5.12), (5.22) and (5.23). The red circles in the last figure show the Sabino bound, which predicts that the regular part of the pair is of dimension 45, much larger than the true order, 4. Indeed, we notice a steep drop in the singular values at index 5.

Applying the GUPTRI software to the Loewner pencil constructed using the com-



(a) Original system (b) Singular value drop

Figure B.1: Original system and singular value drop of the Loewner matrix pencil

plex approach reveals the fact that the regular part is of dimension 4 and the poles are also accurately recovered as the eigenvalues of the top right 4×4 sub-pencil.

```
>> [Sc,Tc,Pc,Qc,kstrc]=guptri(sLL,LL);
>> kstrc
kstrc =
    96    -1    96    -1     4    -1
     0    -1     0    -1     4    -1
>> eig(Sc(1:4,97:100),Tc(1:4,97:100))
ans =
     -9.999999999999869e-01 + 1.000000000000006e+00i
     -9.999999999999996e-01 - 9.999999999999963e-01i
     -2.999999999999900e+00 - 8.267351232444181e-14i
     -5.000000000000414e+00 + 5.865752240334634e-14i
```

The structure given by GUPTRI is divided into 3 parts: the left singular part, the right singular part and the finite part, separated by columns of -1. In this example, the pencil has a left and right singular part of dimension $96 - 0 = 96$, while the dimension of the regular part is 4.

Using the real approach of constructing the matrices, we obtain that the singular part is of dimension 96, while the finite part is of dimension 4, just as it was expected. The poles are also recovered as the eigenvalues of the top right 4×4 sub-pencil.

```
>> [Sr,Tr,Pr,Qr,kstrr]=guptri(sLLN,LLN);
>> kstrr
```

Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	0.062	2.3541e-015	6.9919e-030
1 with SVD	0.078	1.9166e-015	2.8882e-030
1 with GUPTRI	0.16	5.7401e-015	3.7711e-029
2 with random	0.031	2.8360e-015	6.4683e-030
2 with svd	0.078	1.2693e-015	1.2135e-030
2 with GUPTRI	0.163	4.8433e-015	1.8535e-029
3	0.062	1.9716e-015	2.2264e-030
4	0.078	6.2382e-015	6.1345e-029
5	0.062	2.4758e-015	5.1545e-030
6	0.062	7.6516e-015	1.1404e-028
7	0.062	2.4758e-015	5.1545e-030
8	0.046	7.6516e-015	1.1404e-028
9	0.062	2.5994e-015	5.6247e-030
10	**0.078**	**8.0037e-016**	**1.3107e-030**
11	0.078	1.0078e-015	1.5151e-030
12	0.062	1.6631e-015	2.4620e-030
13	0.062	4.3227e-014	5.4518e-027
14	0.062	2.4758e-015	5.1545e-030
15	0.078	7.6516e-015	1.1404e-028
16	0.046	2.4758e-015	5.1545e-030
17	0.078	7.6516e-015	1.1404e-028
18	0.062	1.8472e-015	3.5938e-030
19	0.062	2.8851e-015	3.9217e-029
20	0.078	1.0078e-015	1.5151e-030
21	0.062	7.5753e-015	6.4673e-029
22	0.078	2.2422e-015	3.8968e-030
23	0.062	7.5753e-015	6.4673e-029
24	0.015	1.9982e-014	3.7820e-028
VF	**0.421**	**5.8738e-013**	**3.0046e-025**

Table B.1: Results for $N_s = 100$ noise-free measurements of an order 4 SISO system

```
kstrr =
    96    -1    96    -1     4    -1
     0    -1     0    -1     4    -1
>> eig(Sr(1:4,97:100),Tr(1:4,97:100))
ans =
     -1.000000000000010e+00 + 9.999999999999973e-01i
     -1.000000000000011e+00 - 9.999999999999972e-01i
     -2.999999999999872e+00 + 5.746388301481163e-15i
     -5.000000000000333e+00 - 5.513384913520155e-15i
```

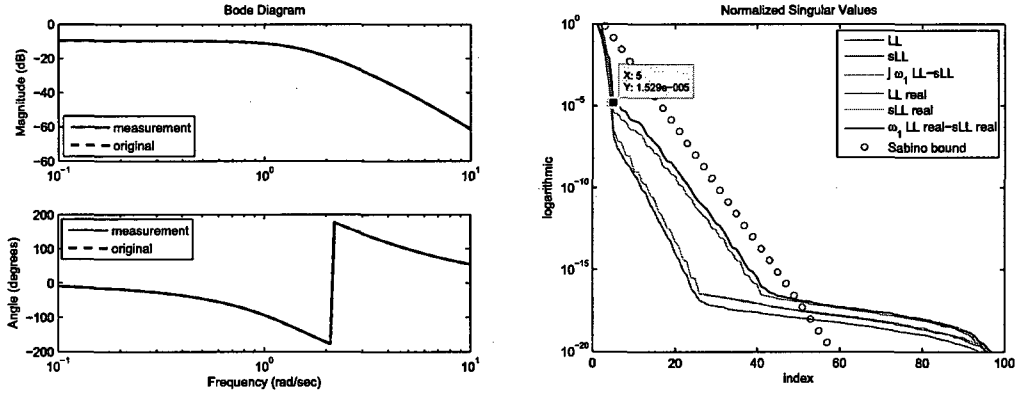
Table B.1 presents the CPU time required by each algorithm to construct an order

4 model, as well as the normalized $\mathcal{H}_\infty$ and $\mathcal{H}_2$ errors for the resulting systems.

We conclude that all the algorithms we proposed were able to recover the original system, yielding errors in the order of machine precision in under 0.1s, while vector fitting took 0.4s and the resulting errors were the highest.

B.2 Noisy SISO system with 4 poles

Consider the same single-input single-output bounded-real system as in the previous section. We sample the transfer function at 100 linearly spaced frequency points between 10^{-1} rad/sec and 10^{1} rad/sec but we assume that we only know the first 6 digits of each measurement. This approach of adding noise to the data is the closest to the real-world scenario of being provided with measurements which have only a certain number of digits after the decimal point. We compare all the algorithms previously proposed in terms of computational time and normalized errors.



(a) Bode Plot

(b) Singular Value Drop

Figure B.2: Bode plot of the system and singular value drop of the Loewner matrix pencil

Fig. B.2(a) shows the Bode plot over the frequency band of interest. The continous line represents the noisy measurement and the interrupted line shows the original value of the transfer function. The two lines are barely noticeable due to the small amount of noise which was introduced. Fig. B.2(b) shows the plot of the normalized singular values of the Loewner and shifted Loewner matrices constructed both using the complex as well as the real approach, with the sampling tangential directions chosen as outlined in Eq. (5.11), (5.12), (5.22) and (5.23). The red circles in the last figure show the Sabino bound, which bounds the drop of the singular values in the real approach quite closely. Recall that the bound cannot be applied to the real approach,

so the approximated bound should be compared to the actual drop of the singular values in the complex approach. It predicts that the regular part of the Loewner pair is of order 45, much larger than the true order, 4. We still notice a steep drop in the singular values at index 5, but the drop is only 5 orders of magnitude (about the amount of noise which was introduced) compared to 16 orders of magnitude in the noise-free case.

Applying GUPTRI with the default values of the input variables EPSU, GAP and ZERO to the Loewner pencil constructed using the complex approach reveals that the finite part is of dimension 4 and the poles, computed as the eigenvalues of the top right 4×4 sub-pencil, are listed below.

```
>> [Sc,Tc,Pc,Qc,kstrc]=guptri(sLL,LL);
>> kstrc
kstrc =
    96    -1    96    -1     4    -1
     0    -1     0    -1     4    -1
>>  eig(Sc(1:4,97:100),Tc(1:4,97:100))
ans =
     -9.999984952259148e-01 + 1.000064552137255e+00i
     -9.999985737335415e-01 - 1.000064532657865e+00i
     -2.996186517891827e+00 - 1.184498083885183e-06i
     -5.015001240950208e+00 + 1.736906966162078e-06i
```

Changing the value of the variable EPSU, namely the uncertainty in the data, to 10^{-6}, the noise level which we introduced, reveals a left and right singular part of dimension $96 - 0 = 96 = 97 - 1$, a finite part of dimension 3, as well as an infinite eigenvalue $(1 - 0 = 1)$. The poles are computed as the eigenvalues of the top right 4×4 sub-pencil.

```
>> [Sc,Tc,Pc,Qc,kstrc]=guptri(sLL,LL,10^(-6));
>> kstrc
kstrc =
    96    -1    97     0    -1     3    -1
     0    -1     1     0    -1     3    -1
>> eig(Sc(1:4,97:100),Tc(1:4,97:100))
ans =
```

```
  -0.9913 - 1.0140i
  -2.4537 + 0.0000i
  -0.9913 + 1.0140i
      Inf
```

Using the real approach of constructing the matrices and the default values of the variables, we obtain that there is no finite part.

```
>> [Sr,Tr,Pr,Qr,kstrr]=guptri(sLLN,LLN);
>> kstrr
kstrr =
    96     0    -1    95     1     1     1     1    -1    -1
     1     0    -1     1     1     1     1     0    -1    -1
```

Changing the value of the variable EPSU, namely the uncertainty in the data, to 10^{-6}, the noise level which we introduced, reveals a finite part of dimension 3 and an infinite eigenvalue.

```
>> [Sr,Tr,Pr,Qr,kstrr]=guptri(sLLN,LLN,10^(-6));
>> kstrr
kstrr =
    96    -1    97     0    -1     3    -1
     0    -1     1     0    -1     3    -1
```

Changing the value of the variable EPSU, namely the uncertainty in the data, to 10^{-7} reveals a finite part of dimension 4, as well as a zero eigenvalue. The poles are computed as the eigenvalues of the top right 5×5 sub-pencil.

```
>> [Sr,Tr,Pr,Qr,kstrr]=guptri(sLLN,LLN,10^(-7));
>> kstrr
kstrr =
    96     0    -1    95    -1     4    -1
     1     0    -1     0    -1     4    -1
>> eig(Sr(1:5,96:100),Tr(1:5,96:100))
ans =
                          0
     -1.001461610150704e+00 - 1.000010623362765e+00i
     -1.001461610150706e+00 + 1.000010623362765e+00i
     -2.981404779714172e+00 - 2.905457961852646e-15i
     -5.030291438781755e+00 + 2.113558320433366e-15i
```

Remark. From the experiments listed above, we conclude that even though we used the value for the uncertainty in the data equal to the amount of noise introduced in the measurements, we did not obtain the results which we expected, namely the true dimension of the finite part of the matrix pencil. One needs to try several values until the desired result is obtained. However, when dealing with true measurements, there is no a-priori insight into what the order of the underlying system is, so the system which GUPTRI outputs will provide to be a poor interpolant due to the fact that the finite part of the system is completely wrong.

Table B.2 presents the CPU time required by each algorithm to construct an order 4 model, as well as the normalized $\mathcal{H}_\infty$ and $\mathcal{H}_2$ errors for the resulting systems. The errors should be compared to the $\mathcal{H}_\infty$ and $\mathcal{H}_2$ errors obtained from comparing the noise-free values to the noisy measurements: the $\mathcal{H}_\infty$ error is $3.8176e-006$, while the $\mathcal{H}_2$ error is $5.0389e-011$.

We notice that the errors in the table are at most 2 orders of magnitude larger for the $\mathcal{H}_\infty$-error and at most 4 orders of magnitude larger for the $\mathcal{H}_2$-error, when compared to the reference errors of $3.8176e-006$ for the $\mathcal{H}_\infty$ error and $5.0389e-011$ for the $\mathcal{H}_2$ error. The smallest errors were obtained from constructing the Loewner matrices in the complex approach using Eq. (5.11), (5.12) and applying GUPTRI on the resulting singular pencil. As before, vector fitting takes more time to produce the results compared to all our algorithms.

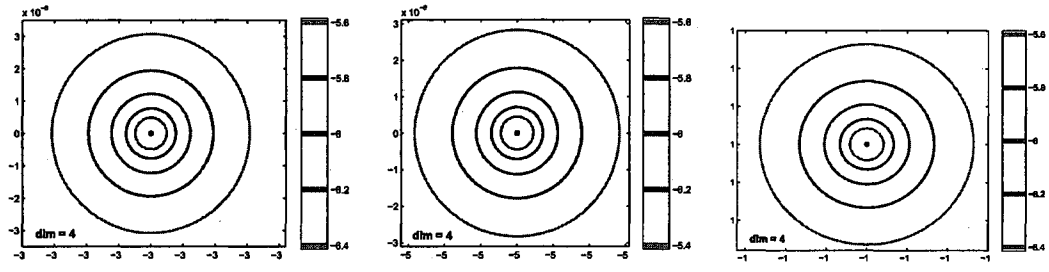
The poles which we recovered using the algorithms presented above are all enclosed in a ball of radius 10^{-6}, as predicted from computing the pseudospectra [26] corresponding to 10^{-6} perturbations applied to the eigenvalues of the $\mathbf{A}$ matrix obtained

Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	0.062	6.8943e-006	6.3849e-011
1 with SVD	0.047	4.8386e-006	4.3336e-011
1 with GUPTRI	**0.147**	**4.8363e-006**	**4.3292e-011**
2 with random	0.078	2.0751e-004	1.7308e-008
2 with SVD	0.062	1.3414e-004	7.7555e-009
2 with GUPTRI	0.216	6.7775e-004	3.7681e-007
3	0.047	1.0830e-005	2.5965e-010
4	0.093	1.4802e-004	4.6331e-008
5	0.047	1.0830e-005	2.5965e-010
6	0.062	1.4802e-004	4.6331e-008
7	0.078	1.0830e-005	2.5965e-010
8	0.078	1.4802e-004	4.6331e-008
9	0.109	1.0830e-005	2.5965e-010
10	0.109	1.1959e-005	2.9920e-010
11	0.093	7.6167e-006	8.5655e-011
12	0.047	1.0830e-005	2.5965e-010
13	0.093	1.4802e-004	4.6331e-008
14	0.078	1.0830e-005	2.5965e-010
15	0.125	1.4802e-004	4.6331e-008
16	0.046	1.0830e-005	2.5965e-010
17	0.078	1.4802e-004	4.6331e-008
18	0.078	1.0830e-005	2.5965e-010
19	0.062	7.2401e-006	1.0378e-010
20	0.015	7.6167e-006	8.5655e-011
21	0.062	9.2169e-006	9.9503e-011
22	0.109	2.7566e-005	4.9260e-010
23	0.047	4.4764e-005	4.2480e-009
24	0.093	1.4953e-005	3.2199e-010
VF	**0.437**	**1.0934e-005**	**7.3038e-011**

Table B.2: Results for $N_s = 100$ noisy measurements of an order 4 SISO system

from a realization of the transfer function of the system:

$$\mathbf{A}_{ss} = \begin{bmatrix} -1 & 1 & 0 & 0 \\ -1 & -1 & 1 & 0 \\ 0 & 0 & -3 & 1 \\ 0 & 0 & 0 & -5 \end{bmatrix}, \mathbf{B}_{ss} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 4 \end{bmatrix}, \mathbf{C}_{ss} = \begin{bmatrix} 2.5 & 0 & 0 & 0 \end{bmatrix}, \mathbf{D}_{ss} = 0. \tag{B.1}$$



(a) Pseudospectra of -3 (b) Pseudospectra of -5 (c) Pseudospectra of $-1 \pm j$

Figure B.3: Pseudospectra of the poles

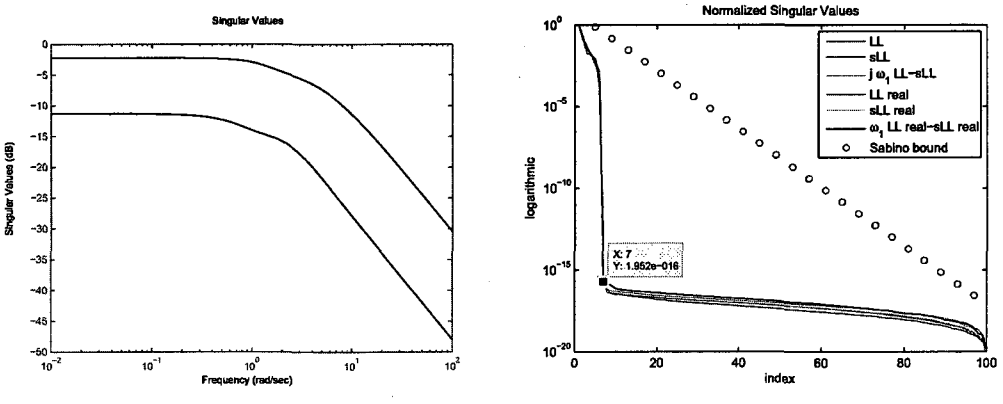
B.3 Noise-free MIMO system with 2 ports and 6 poles

Consider the multiple-input multiple-output (MIMO) bounded-real system given by the matrix transfer function

$$\mathbf{H}(s) = \begin{bmatrix} \frac{1}{2.5(s+1)} & \frac{1}{1.3[(s+1)^2+1]} \\ \frac{1}{[(s+2)^2+3^2]} & \frac{3}{s+5} \end{bmatrix}.$$

The system is of order 6 with poles at $-1 \pm j$, $-2 \pm 3j$, -1 and -5. We sample the transfer function at 100 linearly spaced frequency points between 10^{-1} rad/sec and 10^{1} rad/sec. We compare all the algorithms previously proposed in terms of computational time and normalized errors.

Fig.B.4(a) shows the Bode plot of the system, while Fig.B.4(b) shows the plot of the normalized singular values of the Loewner and shifted Loewner matrices constructed both using the complex as well as the real approach. The red circles in the last figure show the Sabino bound, which predicts that the regular part of the pair is of order 90, much larger than the true order, which is 6. Indeed, we notice a steep drop in the singular values at index 7.



(a) Original system (b) Singular value drop

Figure B.4: Original system and singular value drop of the Loewner matrix pencil

Applying the GUPTRI software to the Loewner pencil constructed using the complex approach reveals that the pencil has a left and right singular part of dimension $94 - 0 = 94$, while the regular part is of dimension 6. The poles are accurately recovered as the eigenvalues of the top right 6×6 sub-pencil.

```
>> [Sc,Tc,Pc,Qc,kstrc]=guptri(sLL,LL,eps);
>> kstrc
kstrc =
    94    -1    94    -1     6    -1
     0    -1     0    -1     6    -1
>> eig(Sc(1:6,95:100),Tc(1:6,95:100))
ans =
     -1.000000000000000e+00 + 6.555344627120030e-16i
     -9.999999999999967e-01 - 9.999999999999996e-01i
     -1.000000000000001e+00 + 1.000000000000002e+00i
     -2.000000000000018e+00 - 2.999999999999996e+00i
     -1.999999999999989e+00 + 3.000000000000017e+00i
     -4.999999999999996e+00 + 6.272943019643350e-15i
```

Using the real approach of constructing the matrices, we obtain that the singular part is of dimension 94, while the regular part is of dimension 6, just as it was expected. The poles are recovered as the eigenvalues of the top right 6×6 sub-pencil.

```
>> [Sr,Tr,Pr,Qr,kstrr]=guptri(sLLN,LLN,eps);
>> kstrr
kstrr =
    94    -1    94    -1     6    -1
     0    -1     0    -1     6    -1
>> eig(Sr(1:6,95:100),Tr(1:6,95:100))
ans =
     -9.999999999999920e-01 + 3.941962367876151e-15i
     -1.000000000000013e+00 - 9.999999999999932e-01i
     -1.000000000000018e+00 + 9.999999999999925e-01i
     -2.000000000000008e+00 + 2.999999999999989e+00i
     -2.000000000000021e+00 - 3.000000000000004e+00i
     -5.000000000000013e+00 + 3.499849599807928e-15i
```

Table B.3 presents the CPU time required by each algorithm to construct an order 6 model, as well as the normalized $\mathcal{H}_\infty$ and $\mathcal{H}_2$ errors for the resulting systems.

Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	0.062	2.3526e-015	4.0734e-030
1 with GUPTRI	0.193	3.0820e-015	1.1539e-029
1 with SVD	0.093	3.4823e-014	3.3014e-028
2 with random	0.093	2.4417e-014	8.5594e-029
2 with GUPTRI	0.16	2.7386e-014	1.7913e-028
2 with SVD	0.078	1.7546e-013	6.3502e-027
3	0.062	8.2669e-015	3.0366e-029
4	0.125	6.9030e-014	3.9088e-027
5	0.046	1.6169e-015	1.1994e-030
6	0.031	7.3394e-014	5.7100e-027
7	0.031	1.4225e-014	1.1251e-028
8	0.062	2.5777e-013	8.8361e-027
9	0.046	1.0165e-014	3.4560e-029
10	0.062	2.3838e-015	7.4356e-030
11	**0.031**	**1.1461e-015**	**1.6672e-030**
12	0.046	3.7240e-014	4.9933e-028
13	0.078	1.4474e-012	6.5960e-025
14	0.031	2.3559e-015	2.1074e-030
15	0.062	6.3334e-014	2.5899e-027
16	0.046	7.0763e-015	3.2192e-029
17	0.109	2.5777e-013	8.8361e-027
18	0.015	1.9238e-014	2.9029e-028
19	0.062	8.3996e-015	9.8826e-029
20	0.046	1.3708e-015	1.4710e-030
21	0.046	3.7726e-014	1.8906e-027
22	0.062	6.7016e-014	1.2545e-027
23	0.062	1.5199e-014	6.3417e-029
24	0.062	8.1214e-014	1.7216e-027
VF	**0.483**	**9.5467e-015**	**6.5172e-029**

Table B.3: Results for $N_s = 100$ noise-free measurements of an order 6 MIMO system

We conclude that all the algorithms tested were able to recover the original system, yielding errors in the order of machine precision in under or around 0.1s, while vector fitting took 0.4s.

B.4 Noisy MIMO system with 2 ports and 6 poles

Consider the same MIMO bounded-real system as in the previous section. We sample the transfer function at 100 linearly spaced frequency points between 10^{-1} rad/sec and 10^{1} rad/sec but we assume that we only know the first 4 digits of the measurement.

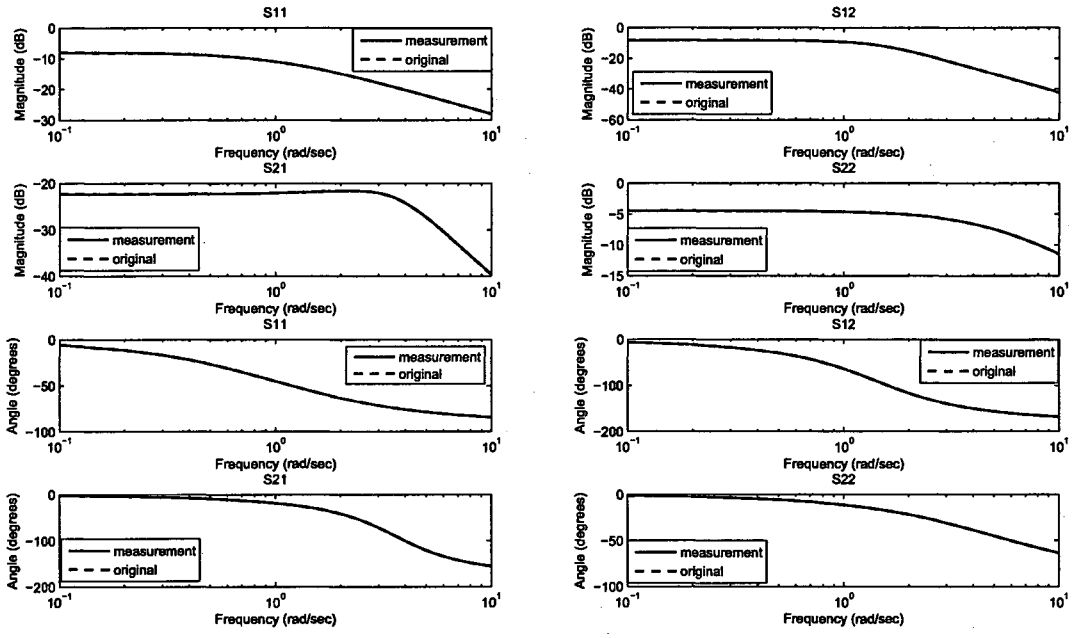
Fig. B.5(a) shows the Bode plot over the frequency band of interest. The continous lines represent the noisy measurements and the interrupted lines show the original value of the transfer function. The two lines are barely noticeable due to the small amount of noise which was introduced. Fig. B.5(b) shows the plot of the normalized singular values of the Loewner and shifted Loewner matrices constructed both using the complex as well as the real approach. The red circles in the last figure show the Sabino bound, which bounds the drop of the singular values in the real approach quite closely. Recall that the bound cannot be applied to the real approach, so the approximated bound should be compared to the actual drop of the singular values in the complex approach. It predicts that the regular part of the Loewner pair is of dimension 90, much larger than the true order, 6. We still notice a steep drop in the singular values at index 7, but the drop is only 4 orders of magnitude (about the amount of noise which was introduced) compared to 16 orders of magnitude in the noise-free case.

Applying GUPTRI with the default values of the variables `EPSU`, `GAP` and `ZERO` to the Loewner pencil constructed using the complex approach gives no finite part.

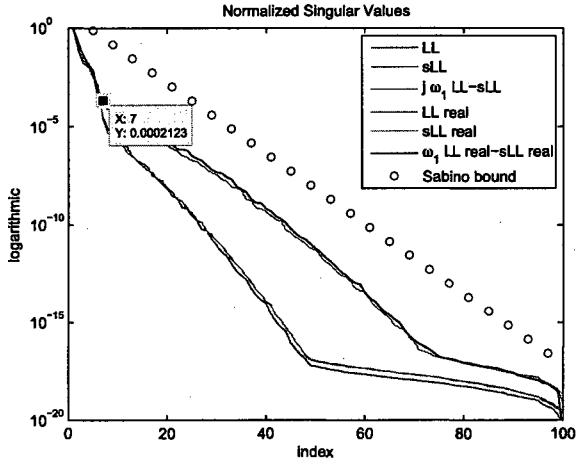
```
>> [Sc,Tc,Pc,Qc,kstrc]=guptri(sLL,LL);
>> kstrc
kstrc =
93     0    -1    92     1     1     1     1     1     1    -1    -1
 2     0    -1     2     1     1     1     1     1     0    -1    -1
```

If we change the uncertainty in the data `EPSU` to 10^{-4}, namely the noise level introduced, we still obtain no finite part.

```
>> [Sc,Tc,Pc,Qc,kstrc]=guptri(sLL,LL,10^(-4));
```



(a) Bode plot



(b) Singular value drop

Figure B.5: Bode plot of the system and singular value drop of the Loewner matrix pencil

```
>> kstrc
kstrc =
    98    -1    98     1     1    -1    -1
     0    -1     1     1     0    -1    -1
```

If we change the uncertainty in the data EPSU to 10^{-6}, we obtain that the finite

part is of dimension 6 and the poles of the system are obtained as the eigenvalues of the 6×6 top right matrix sub-pencil.

```
>> [Sc,Tc,Pc,Qc,kstrc]=guptri(sLL,LL,10^(-6));
>> kstrc
kstrc =
    94    -1    94    -1     6    -1
     0    -1     0    -1     6    -1
>> eig(Sc(1:6,95:100),Tc(1:6,95:100))
ans =
     -9.954153194467835e-01 + 5.310191299049883e-03i
     -9.951650880660542e-01 - 1.002278465082964e+00i
     -9.965191580697733e-01 + 1.005339020459879e+00i
     -1.980045876644297e+00 - 2.975536762760236e+00i
     -1.991230905037634e+00 + 2.980816691279976e+00i
     -4.985875776577374e+00 - 8.926438404988680e-03i
```

If we use the real approach of constructing the matrices and try *all possible combinations of values* for the three parameters `EPSU`, `GAP` and `ZERO`, we are not able to obtain any answer for which the finite part is of dimension 6. We obtained answers with the dimension of the regular part of 0, 1, 2, 3, 4, 5, 15 or 24.

Remark. This simple example of a sixth-order system to which we add a noise level of 10^{-4} shows that GUPTRI does not always output the expected dimension of the regular part of the singular pencil when data is corrupted by noise.

Table B.4 presents the CPU time required by each algorithm to construct an order 6 model, as well as the normalized $\mathcal{H}_\infty$ and $\mathcal{H}_2$ errors for the resulting systems. The errors should be compared to the $\mathcal{H}_\infty$ and $\mathcal{H}_2$ errors obtained from comparing the noise-free values to the noisy measurements: the $\mathcal{H}_\infty$ error is $2.8465e-004$, while the $\mathcal{H}_2$ error is $1.1115e-007$.

We notice that the errors in the table are at most 2 orders of magnitude larger for the $\mathcal{H}_\infty$-error and at most 4 orders of magnitude larger for the $\mathcal{H}_2$-error, when compared to the reference errors of $2.8465e-004$ for the $\mathcal{H}_\infty$ error and $1.1115e-007$ for the $\mathcal{H}_2$ error. The smallest errors were obtained from constructing the Loewner matrices in the complex approach and applying GUPTRI on the resulting singular

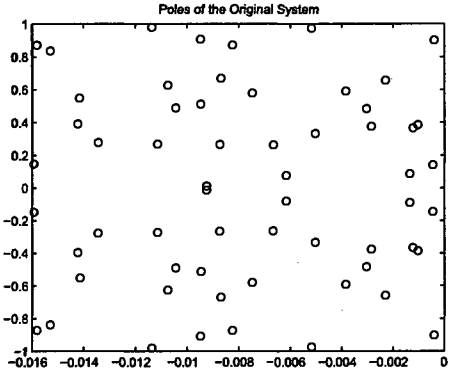
Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	0.015	5.9298e-004	2.4252e-007
1 with GUPTRI	**0.143**	**4.6888e-004**	**2.0460e-007**
1 with svd	0.046	1.3502e-002	3.5758e-005
2 with random	0.062	9.6992e-002	1.6956e-003
2 with svd	0.062	1.6460e-001	1.0714e-002
3	0.062	3.2446e-003	6.7359e-006
4	0.062	5.8039e-003	2.6668e-005
5	0.093	7.9266e-004	4.9781e-007
6	0.078	2.1742e-002	3.2934e-004
7	0.046	9.5423e-003	1.3067e-004
8	0.046	5.4387e-002	2.4011e-003
9	0.046	1.4056e-003	2.5806e-006
10	0.078	1.4390e-003	1.2115e-006
11	0.078	5.6230e-004	3.1693e-007
12	0.015	4.7557e-003	2.5437e-005
13	0.078	8.6766e-002	4.5143e-003
14	0.015	8.6424e-004	5.9844e-007
15	0.125	3.0982e-002	8.4560e-004
16	0.046	1.4782e-003	3.2858e-006
17	0.109	5.4387e-002	2.4011e-003
18	0.062	2.0152e-002	1.5557e-004
19	0.062	2.4492e-003	7.4860e-006
20	0.046	6.3130e-004	3.0616e-007
21	0.046	1.2698e-003	7.6734e-007
22	0.109	3.1688e-002	3.2107e-004
23	0.078	9.2559e-004	7.1200e-007
24	0.078	2.6340e-002	1.7873e-004
VF	**0.421**	**4.2850e-004**	**1.1379e-007**

Table B.4: Results for $N_s = 100$ noisy measurements of an order 6 MIMO system

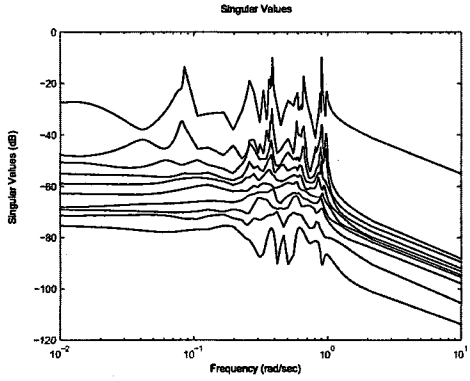
pencil. As before, vector fitting takes more time to produce the results compared to all our algorithms.

B.5 Noise-free system with 10 ports and 60 poles

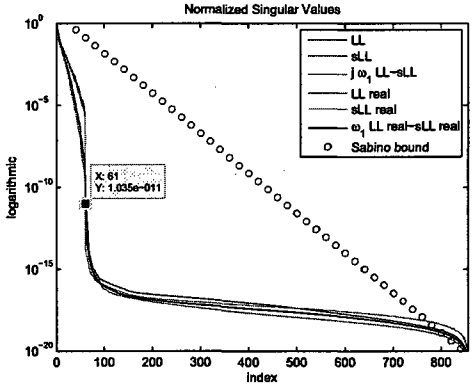
Consider a bounded-real system of order $k = 60$ with $N_p = 10$ input and output ports and poles close to the imaginary axis (Fig. B.6(a)).



(a) Poles



(b) Original system



(c) Singular value drop

Figure B.6: Poles, sigma plot and singular value drop of the Loewner matrix pencil

We sample the transfer function at 856 frequency points between 10^{-2} rad/sec and 10^{1} rad/sec (the plot of singular values of the transfer function is presented in Fig. B.6(b)). We compare all the algorithms previously proposed in terms of computational time and normalized errors in trying to recover the original system.

Fig. B.6(c) shows the plot of the normalized singular values of the Loewner and shifted Loewner matrices constructed both using the complex as well as the real approach. The red circles in the last figure show the Sabino bound, which predicts

that dimension of the regular part of the Loewner the pair is 600, much larger than the true order, 60.

The purpose of this example is to test the ability of the algorithms to recover the original system when noise-free measurements of a system with a large number of ports and a large order are taken. This system was randomly generated and was chosen such that it is close to the real-world systems in terms of large number of ports and large number of samples. Moreover, by choosing the poles close to the imaginary axis, the frequency response of the system will be harder to approximate due to the numerous spikes on the plot.

Applying the GUPTRI software to the Loewner pencil constructed using the complex approach reveals the fact that the left and right singular part are of dimension $796 - 0 = 796$, while the regular part is of dimension 60.

```
>> [Sc,Tc,Pc,Qc,kstrc]=guptri(sLL,LL,10^(-10));
>> kstrc
kstrc =
   796    -1   796    -1    60    -1
     0    -1     0    -1    60    -1
```

Using the real approach of constructing the matrices and *all possible combinations of the values of the parameters EPSU, GAP and ZERO*, we were not able to recover the finite part of the pencil as expected. Most of the outputs contained no finite part, and the largest we could obtain was 4, using EPSU= 10^{-5}. Given that the measurements are noise-free, the input value of the parameter for the uncertainty in the data should be very small (order of machine precision). However, small values of EPSU lead to no finite part.

Remark. This easy example proves that GUPTRI fails at recovering the regular part of a singular Loewner matrix pencil, even in the case of noise-free measurements.

Table B.5 presents the CPU time required by each algorithm to construct an order 60 model, as well as the normalized $\mathcal{H}_\infty$ and $\mathcal{H}_2$ errors for the resulting systems.

Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	16.427	4.4017e-009	7.4871e-017
1 with SVD	**17.114**	**2.1518e-009**	**5.5949e-018**
1 with GUPTRI	94.837	8.5360e-009	9.5851e-017
2 with random	17.301	3.0092e-003	4.3415e-006
2 with SVD	17.191	1.2962e-003	6.9737e-007
3	17.769	7.2281e-006	1.8563e-011
4	9.4225	8.2998e-005	4.2078e-009
5	18.049	4.7703e-005	5.7159e-010
6	9.0169	1.3121e-005	3.1666e-010
7	18.361	2.1775e-007	4.2825e-014
8	9.5941	2.1880e-005	2.9598e-010
9	2.4024	1.9431e-007	4.7815e-014
10	3.4164	4.2529e-009	2.9673e-017
11	3.2916	7.6730e-009	7.9867e-017
12	17.691	6.4849e-005	3.7148e-009
13	9.4069	7.2989e-004	4.9529e-007
14	18.533	8.3472e-005	4.7185e-009
15	9.7033	4.1057e-006	9.3990e-012
16	17.956	4.0174e-007	2.1632e-013
17	9.4849	1.2146e-005	1.5895e-010
18	2.3556	1.3099e-007	3.9977e-014
19	3.4164	1.8858e-008	2.9695e-016
20	**3.4476**	**3.3750e-009**	**2.7644e-017**
21	56.894	3.4995e-007	1.8010e-013
22	26.302	1.8401e-005	5.9306e-010
23	57.252	8.5076e-007	4.3203e-013
24	26.676	2.8808e-004	2.9722e-008
VF	**6.1308**	**2.3595e-001**	**3.9163e-001**

Table B.5: Results for $N_s = 856$ noise-free measurements of an order 60 MIMO system with $N_p = 10$ ports

Table B.5 shows that all the algorithms proposed recovered the original system. The smallest errors were obtained with algorithm 1, but the computational time required is too large. However, algorithm 20 leads to a good model in less than 3.5s. On the other hand, vector fitting produced a model far from the original. Increasing the number of iterations for each column gave errors in the same order of magnitude. The dimension 500 model created with VF still did not produce small errors: the normalized $\mathcal{H}_\infty$ error is $1.1866e-002$, while the normalized $\mathcal{H}_2$ error is $3.2553e-004$.

B.6 Examples obtained from measurements

The data sets were provided by CST Ltd. They were obtained as measurements taken with a vector network analyzer (VNA). There are N_s frequency samples. For each frequency sample a matrix of dimension $N_p \times N_p$ with complex entries, representing the measured S-parameters, is given.

B.6.1 200 measurements from a device with 26 ports

This data set contains $N_s = 200$ frequency samples between 5MHz and 1GHz. In order to avoid numerical instabilities, all frequencies were scaled by 10^{-6}.

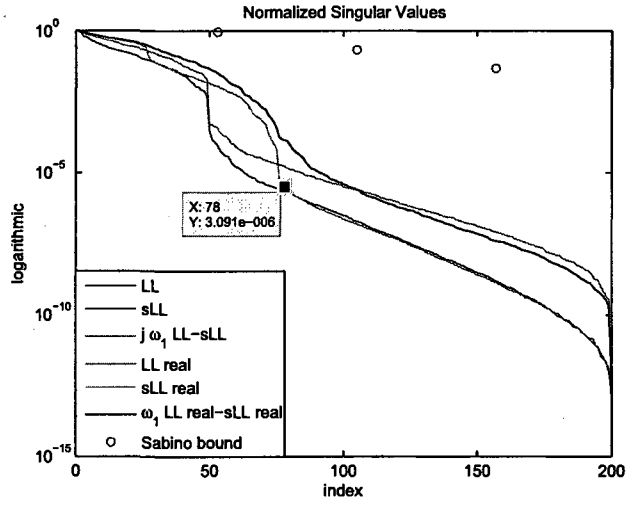


Figure B.7: Singular value drop of the Loewner pencil

First, we construct the big Loewner and shifted Loewner matrices in the real and complex approach using the sampling directions as outlined in Eq. (5.11), (5.12), (5.22) and (5.23). Next, we compute how fast the singular values drop and check whether Sabino's bound is able to match the drop of the normalized singular values. Figure B.7 shows that the predicted decay is much slower than the actual decay.

Applying GUPTRI on the pencil constructed using the complex approach with the uncertainty in the data set to 10^{-14} reveals a finite part of dimension 147:

```
>> [Sc,Tc,Pc,Qc,kstrc]=guptri(sLL,LL,10^(-14));
>> kstrc
kstrc =
Columns 1 through 12
14     9     5     4     4     4     3     2     2     1     1     1
14     9     5     4     4     4     3     2     2     1     1     1
Columns 13 through 19
0     -1     3     0    -1   147    -1
0     -1     3     0    -1   147    -1
```

Applying GUPTRI on the pencil constructed using the real approach with the uncertainty in the data set to 10^{-14} reveals a regular part of dimension 200:

```
>> [Sr,Tr,Pr,Qr,kstrr]=guptri(sLLN,LLN,10^(-14));
>> kstrr
kstrr =
    -1    -1   200    -1
    -1    -1   200    -1
```

Any other value of the uncertainty smaller than 10^{-14} leads to either no finite part in the complex approach, or no finite part in the real approach. Since the data is given in the magnitude-angle format, rather than the real-imaginary format, the accuracy of the measurements could be close to 14 digits.

The results which GUPTRI outputs are in accordance with the behaviour shown in Figure B.7, where the singular values of the pencil constructed using the real approach are decaying slower than the singular values of the pencil construted using the complex approach.

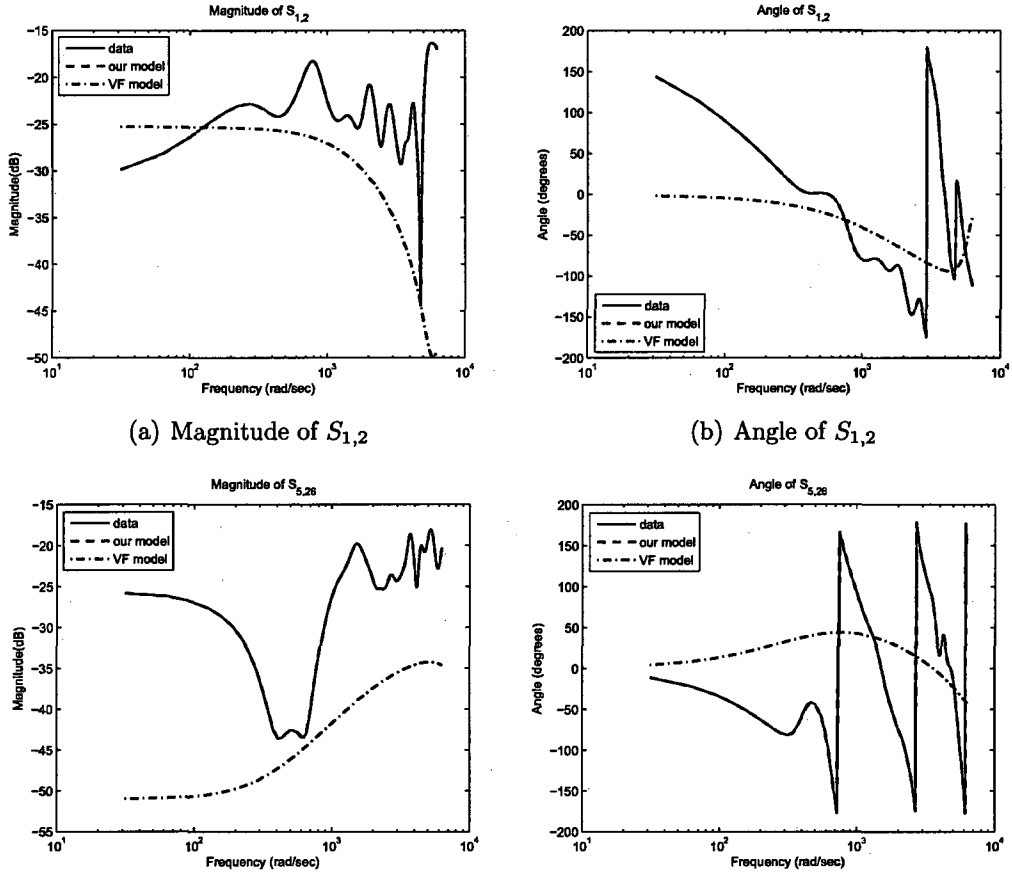
Due to the fact that, to compare models obtained with our algorithms to VF, we need to choose the order of the interpolating system as a multiple of the number of ports, we are presenting the results for interpolants of degree $k = 78$ (choosing $k = 26$ or $k = 26 \cdot 2 = 52$ would lead to bad interpolants) in Table B.6 and Figure B.8. The x-axis of the plots in Figure B.8 have a logarithmic scale and the frequencies are scaled by 10^{-6}. Note that the dimension $k = 78$ ensures that the resulting $(\mathbf{A}, \mathbf{E})$ matrix pencil of the reduced system is not singular.

Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	0.202	1.3453e-001	4.9170e-005
1 with SVD	0.686	1.2164e-001	1.3779e-004
2 with random	0.296	4.2339e-001	1.0673e-003
2 with SVD	**0.858**	**1.6340e-002**	**3.7294e-006**
3	12.62	2.9634e-001	1.3109e-003
4	6.396	4.02	1.7810e-001
5	12.386	1.1098e-001	9.0517e-005
6	6.552	5.0467	2.4311e-002
7	13.026	1.8587e-001	3.1009e-004
8	6.661	23.233	5.1675e-001
9	0.92	3.6352e-001	1.9044e-003
10	1.575	1.4913e-001	2.8680e-004
11	1.934	4.2053e-002	3.9141e-005
12	12.698	3.3578e-001	1.0998e-003
13	6.676	10.414	4.1212e-001
14	12.745	1.0719e-001	1.2003e-004
15	6.817	1.7019	2.7045e-002
16	12.87	4.3380e-001	1.6205e-003
17	7.004	2.478	4.9647e-002
18	0.92	1.329	2.7666e-002
19	1.7	2.3897e-001	9.8438e-004
20	1.934	4.1323e-002	2.6909e-005
21	39.468	2.942	6.5191e-002
22	11.388	1.213	1.3044e-002
23	39.655	4.959	3.5737e-001
24	7.035	4.138	1.5727e-001
VF	**4.5084**	**1.3224**	**1.0546e-001**

Table B.6: Results for constructing a model of dimension $k = 78$ from a data set obtained from a device with $N_p = 26$ ports

Clearly, some of the algorithms proposed, as well as VF, yielded bad models. Some of the computational times were too large, even though the errors were small (for example, algorithms 3, 5, 7, 12, 14, 16). Overall, we conclude that algorithms 1 with random sampling directions, 2 with the sampling directions taken from the SVD of the scattering matrices, 9, 10, 11, 19, 20 provide the best accuracy for a reasonable computational time, with algorithm 2 building the best model.

Vector fitting starts improving its performance, so for an order $k = 780$ model (10 times larger than the dimension of our systems!), the errors are comparable to those



(a) Magnitude of $S_{1,2}$

(b) Angle of $S_{1,2}$

(c) Magnitude of $S_{5,26}$

(d) Angle of $S_{5,26}$

Figure B.9: Comparison of the models built with algorithm 2 and with VF to the data obtained from a device with $N_p = 26$ ports

B.6.2 200 measurements from a device with 16 ports

This data set contains $N_s = 200$ frequency samples between 5MHz and 1GHz. In order to avoid numerical instabilities, all frequencies were scaled by 10^{-6}.

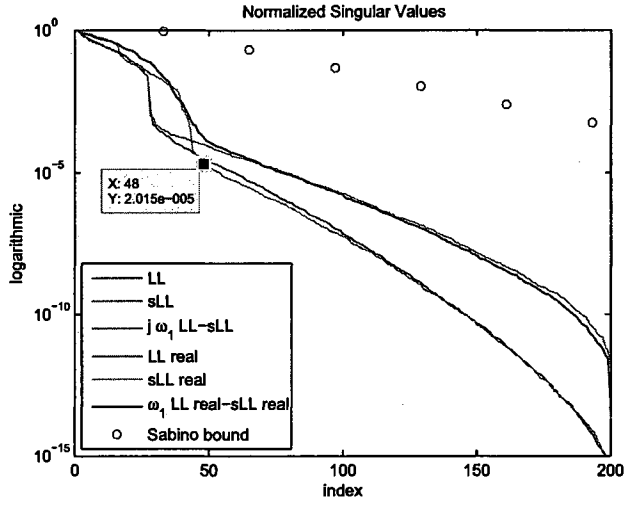


Figure B.10: Singular value drop of the Loewner pencil

First, we construct the big Loewner and shifted Loewner matrices in the real and complex approach using the sampling directions as outlined in Eq. (5.11), (5.12), (5.22) and (5.23). Next, we compute how fast the singular values drop and check whether Sabino's bound is able to match the drop of the normalized singular values. Figure B.10 shows that the predicted decay is much slower than the actual decay.

Applying GUPTRI on the pencil constructed using the complex approach with EPSU set to 10^{-16} and GAP set to 100 reveals a finite part of dimension 101:

```
>> [Sc,Tc,Pc,Qc,kstrc]=guptri(sLL,LL,10^(-16),100);
>> kstrc
kstrc =
Columns 1 through 12
14    10     9     6     4     5     4     4     3     3     3     3
14    10     9     6     4     5     4     4     3     3     3     3
Columns 13 through 24
2     2     2     2     3     2     2     2     1     2     3     2
2     2     2     2     3     2     2     2     1     2     3     2
Columns 25 through 34
```

```
1     1     3     0    -1     1     0    -1   101    -1
1     1     3     0    -1     1     0    -1   101    -1
```

Applying GUPTRI on the pencil constructed with the real approach, the uncertainty set to 10^{-16} and GAP=100 gives a finite part of dimension 166:

```
>> [Sr,Tr,Pr,Qr,kstrr]=guptri(sLLN,LLN,10^(-16),100);
>> kstrr
kstrr =
Columns 1 through 12
2     2     1     3     1     3     2     2     2     2     1     2
2     2     1     3     1     3     2     2     2     2     1     2
Columns 13 through 24
1     2     2     2     2     1     1     0    -1    -1   166    -1
1     2     2     2     2     1     1     0    -1    -1   166    -1
```

Any other combination of the value of the uncertainty smaller than 10^{-16} together with any value of the GAP variable leads to either no finite part in the complex approach, or no finite part in the real approach. Since the data is given in the magnitude-angle format, rather than the real-imaginary format, the accuracy of the measurements could be close to 16 digits.

The results which GUPTRI outputs are in accordance with the behaviour shown in Figure B.10, where the singular values of the pencil constructed using the real approach are decaying slower than the singular values of the pencil construted using the complex approach.

Due to the fact that, to compare models obtained with our algorithms to VF we need to choose the order of the interpolating system as a multiple of the number of ports, we are presenting the results for interpolants of degree $k = 48$ (choosing $k = 16$ or $k = 16 \cdot 2 = 32$ would lead to bad interpolants) in Table B.7 and Figure B.11. Note that the dimension $k = 48$ ensures that the resulting $(\mathbf{A}, \mathbf{E})$ matrix pencil of the reduced system is not singular.

Clearly, some of the algorithms proposed, as well as VF, yielded bad models. Some of the computational times were too large, even though the errors were small (for

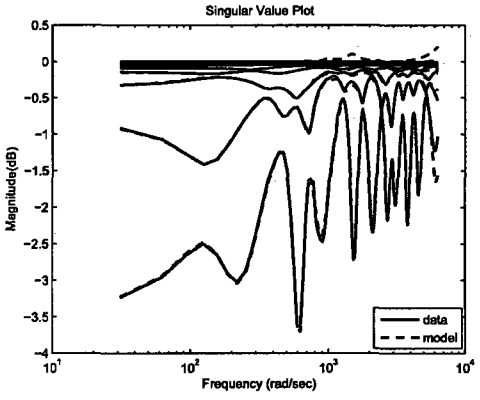
Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	0.218	2.3624e-001	6.8270e-004
1 with SVD	0.39	1.2765e-001	1.4144e-004
2 with random	**0.234**	**3.6503e-002**	**1.0815e-005**
2 with SVD	0.43	5.6151e-002	1.4743e-005
3	2.667	3.2340e-001	2.1318e-003
4	1.419	1.886	4.4858e-002
5	2.901	1.2220e-001	1.9486e-004
6	1.482	2.9015e-001	2.2037e-003
7	2.73	2.3861e-001	7.7106e-004
8	1.7	5.1874e-001	6.0921e-003
9	0.468	3.321	1.5105e-001
10	0.624	3.0645e-001	3.8295e-004
11	0.624	9.6983e-002	2.1237e-004
12	2.761	3.0073e-001	1.7967e-003
13	1.466	1.764	5.5640e-002
14	2.901	9.8385e-002	2.0587e-004
15	1.7	6.0200e-001	4.1206e-003
16	2.636	3.2954e-001	6.9842e-004
17	1.482	2.798	1.1262e-001
18	0.374	4.2948e-001	4.4352e-003
19	0.546	1.9641e-001	7.1721e-004
20	0.561	2.0408e-001	2.5138e-004
21	8.159	16.419	1.08
22	2.464	4.474	1.4187e-001
23	8.268	12.184	1.384
24	12.784	2.694	5.6223e-002
VF	**1.903**	**1.525**	**1.3921e-001**

Table B.7: Results for constructing a model of dimension $k = 48$ from a data set obtained from a device with $N_p = 16$ ports

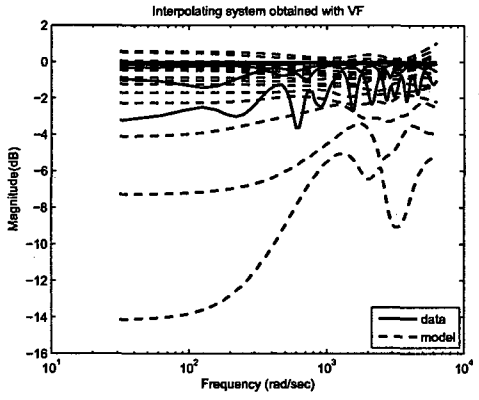
example, algorithms 3, 5, 6, 7, 8, 12, 14, 15, 16). Overall, we conclude that algorithms 1, 2, 10, 11, 18, 19, 20 provide the best accuracy for a reasonable computational time (less than 1s). Algorithm 2 using random sampling directions built the best model.

Vector fitting starts improving its performance, so for an order $k = 384$ model (8 times larger than the dimension of our systems!), the errors are comparable to our algorithms. However, the computational time has also increased to 17.862s.

We also checked how well the magnitude and the angle of some of the entries are modeled in Figure B.12.

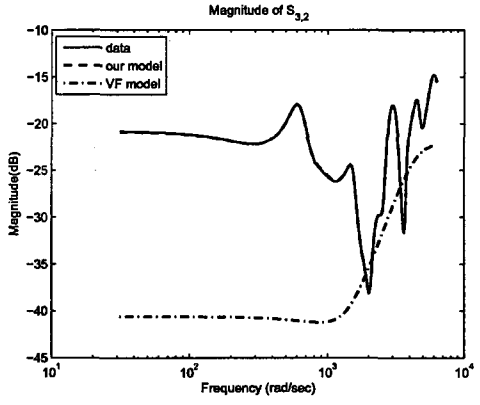


(a) $k = 48$ system with algorithm 2

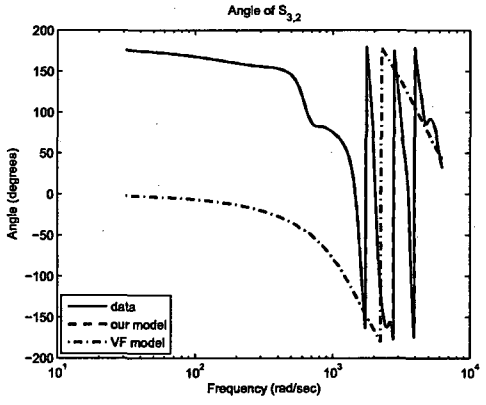


(b) $k = 48$ system with VF

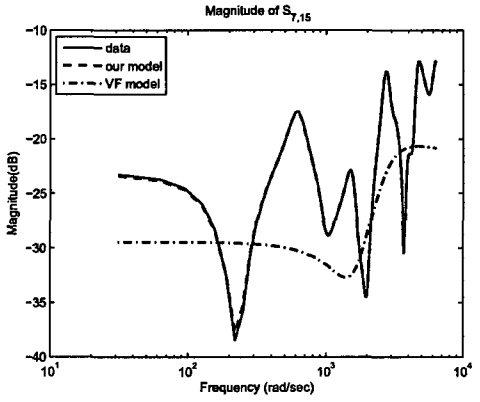
Figure B.11: Models of dimension $k = 48$ obtained with algorithm 2 and with VF for a data set with $N_p = 16$ ports



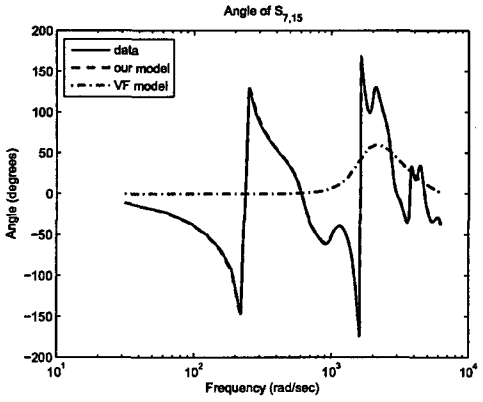
(a) Magnitude of $S_{3,2}$



(b) Angle of $S_{3,2}$



(c) Magnitude of $S_{7,15}$



(d) Angle of $S_{7,15}$

Figure B.12: Comparison of the models built with algorithm 2 and with VF to the data obtained from a device with $N_p = 16$ ports

B.6.3 1000 measurements from a device with 14 ports

This data set contains $N_s = 1000$ frequency samples between 0.01MHz and 10MHz. In order to avoid numerical instabilities, all frequencies were scaled by 10^{-6}.

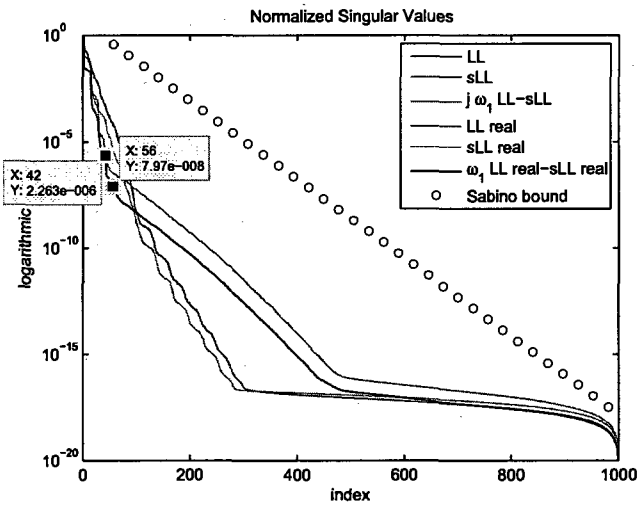


Figure B.13: Singular value drop of the Loewner pencil

First, we construct the big Loewner and shifted Loewner matrices in the real and complex approach using random sampling directions of the left and right tangential interpolation data. Next, we compute how fast the singular values drop and check whether Sabino's bound is able to match the drop of the normalized singular values. Figure B.13 shows that the predicted decay is much slower than the actual decay.

The GUPTRI software takes very long to run on such big matrices (the matrices are of dimension 1000) and we could not obtain an answer even after 2h.

Due to the fact that, to compare models obtained with our algorithms to VF we need to choose the order of the interpolating system as a multiple of the number of ports, we are presenting the results for interpolants of degree $k = 42$ (choosing $k = 14$ or $k = 14 \cdot 2 = 28$ would lead to bad interpolants) in Table B.8 and Figure B.14.

Clearly, some of the algorithms proposed yielded bad models. Some of the computational times were too large, even though the errors were small (for example,

Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	23.213	4.1545e-001	1.5219e-003
1 with SVD	24.445	5.2532e-001	1.9430e-002
2 with random	23.182	1.723	1.7449e-002
2 with SVD	18.330	1.7799	1.5080e-002
3	18.112	2.7407	7.0460e-002
4	8.611	9.806	6.631
5	**10.655**	**2.5114e-001**	**3.2448e-003**
6	6.942	14.763	1.5301e-001
7	14.134	1.472	3.7448e-002
8	7.519	6.704	1.2556e-001
9	1.716	12.463	1.8027e-001
10	3.322	7.5677e-001	6.8955e-002
11	**4.664**	**6.9118e-001**	**4.3446e-002**
12	19.703	48.289	8.4847e-001
13	9.407	127.09	5.848
14	17.644	1.921	1.0965e-001
15	8.096	7.5675e-001	2.8630e-002
16	16.053	8.547	3.9421e-001
17	8.642	63.811	1.289
18	2.09	7.801	2.1666e-001
19	3.291	1.175	1.5142e-001
20	3.151	8.8516e-001	6.7564e-002
21	35.256	1.404	4.8909e-001
22	29.703	46.903	5.4688e-001
23	36.192	2.218	9.1549e-001
24	31.731	17.648	2.9965e-001
VF	**8.299**	**1.178**	**1.3455e-002**

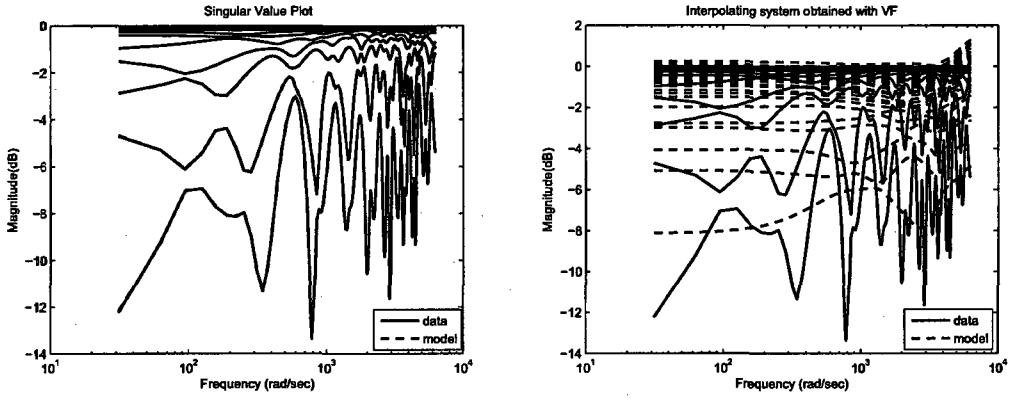
Table B.8: Results for constructing a model of dimension $k = 42$ from a data set obtained from a device with $N_p = 14$ ports

algorithms 1, 15 and 5, the last proving to be the best model). Overall, we conclude that algorithms 10, 11, 20 provide the best accuracy for a reasonable computational time (less than 4s).

Next, let us compare the computational time and the errors when an order $k = 56$ model is generated using all the algorithms. The results are presented in Table B.9 and Figure B.15.

Clearly, some of the algorithms proposed yielded bad models. Overall, we conclude that algorithms 1, 2, 5, 10, 14, 15 and VF are able to construct a good model, with

obtained with our algorithms. However, the computational time has also increased to 60.96s.

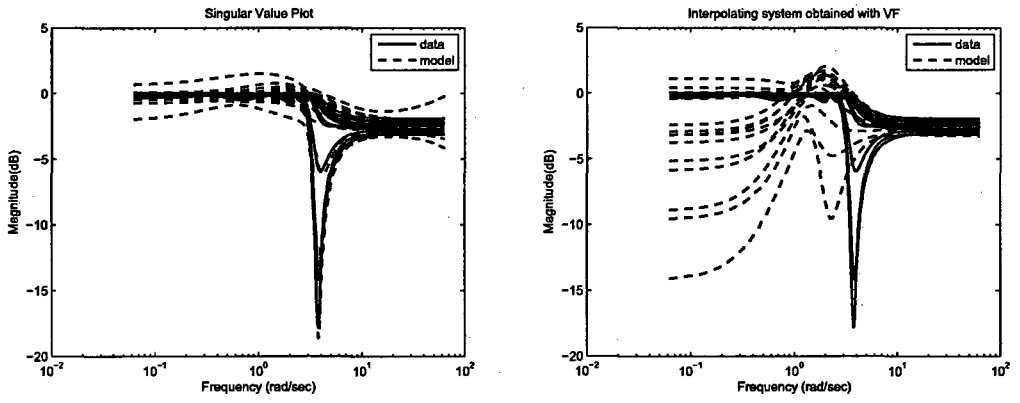


(a) $k = 78$ system with algorithm 2

(b) $k = 78$ system with VF

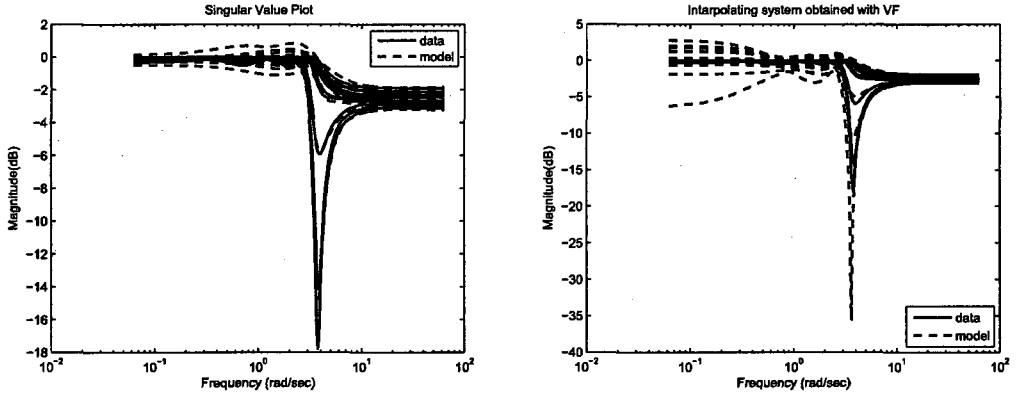
Figure B.8: Models of dimension $k = 78$ obtained with algorithm 2 and with VF for a data set with $N_p = 26$ ports

We also checked how well the magnitude and the angle of some of the entries are modeled. Figure B.9 compares the measured $S_{1,2}$ and $S_{5,26}$ entries to the model obtained with algorithm 2 and to the one obtained with VF. Clearly, our model is hardly distinguishable from the data, while VF is far from good.



(a) $k = 42$ system with algorithm 5 (b) $k = 42$ system with VF

Figure B.14: Models of dimension $k = 42$ built with algorithm 5 and with VF for a data set with $N_p = 14$ ports



(a) $k = 56$ system with algorithm 14 (b) $k = 56$ system with VF

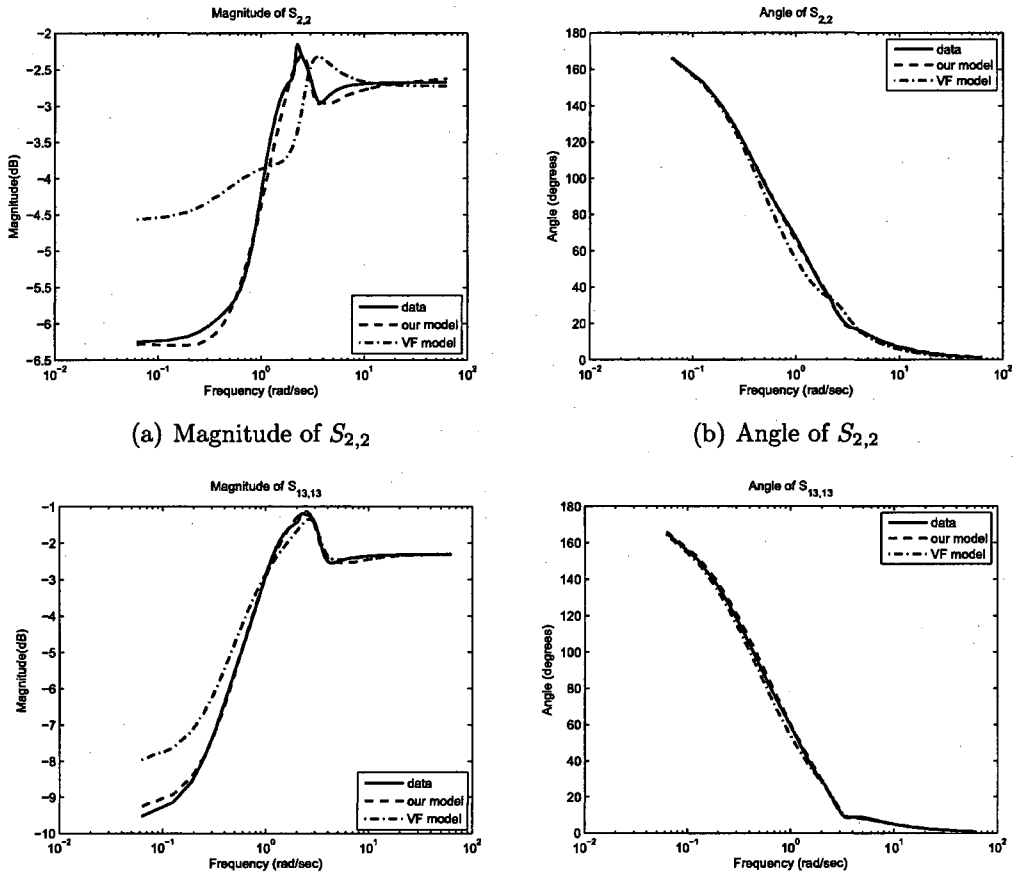
Figure B.15: Models of dimension $k = 56$ obtained with algorithm 14 and with VF for a data set with $N_p = 14$ ports

14 giving the best order 56 model. Algorithm 10 provides the best accuracy for a reasonable computational time (less than 4s).

We also checked how well the magnitude and the angle of some of the entries are modeled. Figure B.16 compares the measured $S_{2,2}$ and $S_{13,13}$ entries to the model obtained with algorithm 14 and to the one obtained with VF. Clearly, our model is hardly distinguishable from the data, while VF is far from good.

Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	24.071	2.9174e-001	4.5044e-004
1 with SVD	23.358	3.2528	1.2098e-002
2 with random	23.478	4.6846e-001	8.0572e-004
2 with SVD	17.735	1.7412	1.1303e-002
3	21.201	4.937	9.7243e-002
4	12.246	10.085	1.5740e-001
5	21.903	3.8383e-001	1.6684e-003
6	10.374	1.325	5.4178e-003
7	20.405	1.376	1.0034e-002
8	10.296	2.905	2.6056e-001
9	2.121	4.756	1.6988e-001
10	**3.728**	**5.3072e-001**	**1.9013e-002**
11	3.884	1.88	2.6304e-002
12	20.561	7.385	1.9185e-001
13	10.327	25.537	5.5401e-001
14	**20.187**	**2.1135e-001**	**8.7416e-004**
15	11.357	8.7690e-001	9.3700e-003
16	21.31	1.308	6.9095e-002
17	10.374	1.503	8.2474e-002
18	1.918	1.666	3.5504e-002
19	3.416	6.3013e-001	4.2986e-002
20	3.4008	8.976	1.3084e-001
21	66.909	3.627	3.4068e-001
22	38.47	6.689	7.2096e-002
23	66.831	9.782	6.4342e-001
24	40.139	36.91	8.0734e-001
VF	**13.198**	**4.9860e-001**	**2.3784e-003**

Table B.9: Results for constructing a model of dimension $k = 56$ from a data set obtained from a device with $N_p = 14$ ports



(a) Magnitude of $S_{2,2}$

(b) Angle of $S_{2,2}$

(c) Magnitude of $S_{13,13}$

(d) Angle of $S_{13,13}$

Figure B.16: Comparison of the models built with algorithm 2 and with VF to the data obtained from a device with $N_p = 14$ ports

B.6.4 1000 measurements from a device with 8 ports

This data set contains $N_s = 1000$ frequency samples between 1MHz and 1GHz. In order to avoid numerical instabilities, all frequencies were scaled by 10^{-9}.

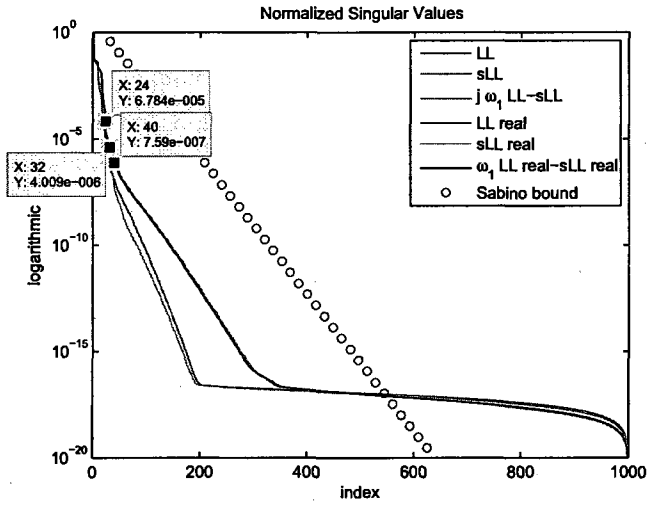


Figure B.17: Singular value drop of the Loewner pencil

First, we construct the big Loewner and shifted Loewner matrices in the real and complex approach using random sampling directions of the left and right tangential interpolation data. Next, we compute how fast the singular values drop and check whether Sabino's bound is able to match the drop of the normalized singular values. Figure B.17 shows that the predicted decay is much slower than the actual decay.

The GUPTRI software takes very long to run on such big matrices (the matrices are of dimension 1000) and we could not obtain an answer even after 2h.

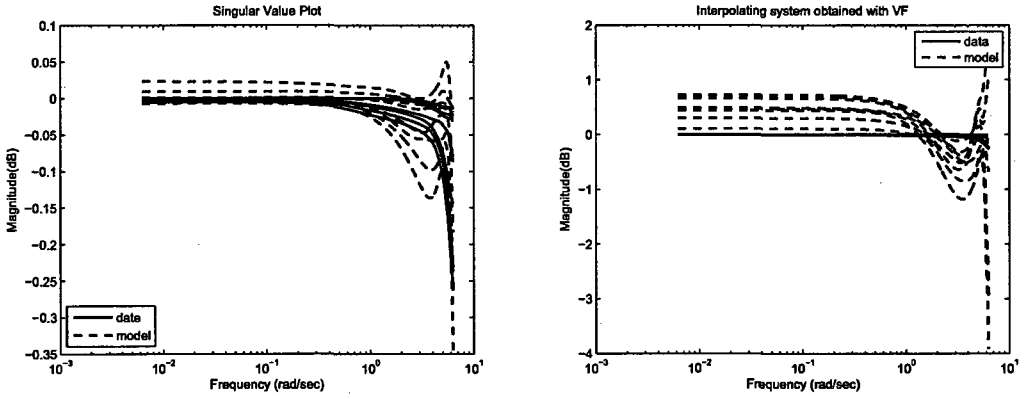
Due to the fact that, to compare models obtained with our algorithms to VF we need to choose the order of the interpolating system as a multiple of the number of ports, we are presenting the results for interpolants of degree $k = 24$ (choosing $k = 8$ or $k = 8 \cdot 2 = 16$ would lead to bad interpolants) in Table B.10 and Figure B.18. Since the singular values are all very close to 0dB, we deduce from Figure B.18 that the underlying device from which the measurements were taken is a loss-free system.

Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	24.009	6.1742e-002	2.8286e-005
1 with SVD	25.272	1.7885e-001	3.0292e-004
2 with random	22.683	5.1503e-002	2.1444e-005
2 with SVD	18.096	7.3070e-002	6.7911e-005
3	2.714	2.4645e-001	2.0742e-003
4	1.544	2.7710e-001	6.1307e-003
5	2.808	1.9213e-002	3.5345e-005
6	1.607	4.1377e-002	1.9567e-004
7	2.948	4.8115e-002	1.6735e-004
8	1.591	1.5078e-001	1.6665e-003
9	0.639	7.1509e-002	3.0212e-004
10	0.967	1.3061e-001	2.5370e-004
11	**1.014**	**1.8995e-002**	**2.7951e-005**
12	2.73	3.7204e-001	2.2304e-003
13	1.575	2.6841e-001	2.7032e-003
14	2.87	1.9767e-002	3.6147e-005
15	1.856	4.2629e-002	2.1024e-004
16	2.901	8.4276e-002	2.4812e-004
17	1.8252	4.6207e-001	4.0190e-003
18	0.515	5.9321e-001	7.7079e-003
19	0.952	7.6071e-002	8.3196e-005
20	0.78	1.9104e-002	3.0952e-005
21	8.486	1.0016e-001	1.5513e-004
22	24.867	1.7488e-001	3.9217e-004
23	8.018	6.5135e-002	1.1631e-004
24	29.656	8.7803e-002	2.0797e-004
VF	**2.496**	**6.3010e-001**	**1.5106e-002**

Table B.10: Results for constructing a macromodel of dimension $k = 24$ from a data set obtained from a device with $N_p = 8$ ports

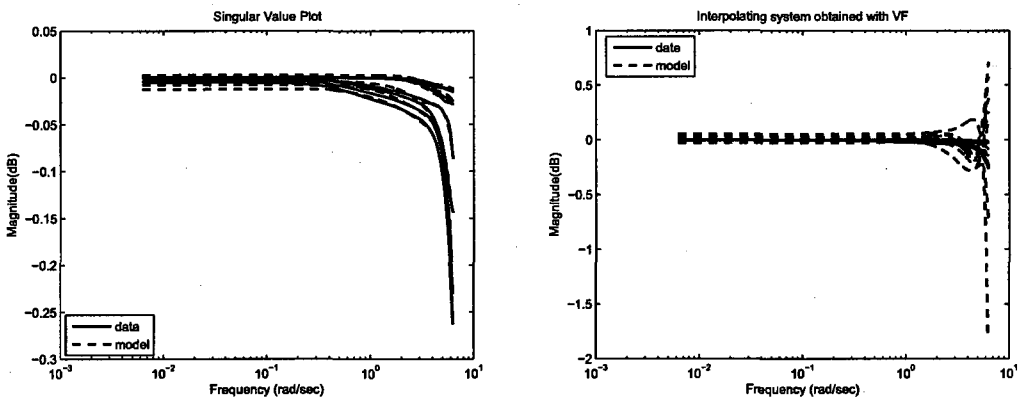
Clearly, even though some of the algorithms proposed yielded good models, the computational times were too large (for example, algorithms 1, 2, 21, 22, 23, 24). Overall, we conclude that algorithms 5, 6, 7, 9, 10, 11, 14, 15, 16, 19 and 20 provide the best accuracy for a reasonable computational time (less than 3s). The system built with algorithm 11 is the best order $k = 24$ model.

Next, let us compare the computational time and the errors when an order $k = 32$ model is generated using all the algorithms. The results are presented in Figure B.19 and Table B.11.



(a) $k = 24$ system with algorithm 11 (b) $k = 24$ system with VF

Figure B.18: Models of dimension $k = 24$ obtained with algorithm 11 and with VF for a data set with $N_p = 8$ ports



(a) $k = 32$ system with algorithm 20 (b) $k = 32$ system with VF

Figure B.19: Models of dimension $k = 32$ obtained with algorithm 20 and with VF for a data set with $N_p = 8$ ports

Clearly, even though some of the algorithms proposed yielded good models, the computational times were too large (for example, algorithms 1, 2, 5, 7, 14, the last giving the best order $k = 32$ model). Overall, we conclude that algorithms 6, 11, 15, 19, 20 provide the best accuracy for a reasonable computational time (less than 3.5s).

Next, let us compare the computational time and the errors when an order $k = 40$ model is generated using all the algorithms. The results are presented in Figure B.20 and Table B.12.

Clearly, even though some of the algorithms proposed yielded good models, the

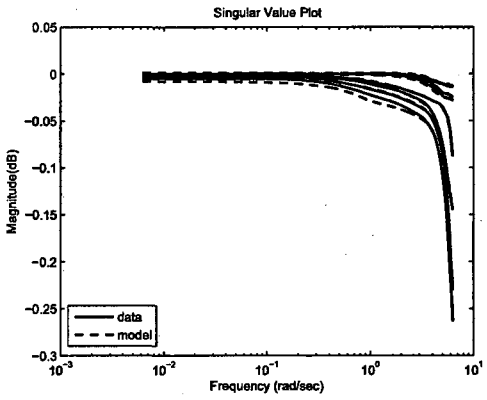
Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	27.207	2.2658e-002	4.6241e-006
1 with SVD	25.319	2.0286e-002	3.8977e-006
2 with random	24.321	2.4272e-002	4.8133e-006
2 with SVD	17.628	7.6629e-002	3.0362e-005
3	5.21	1.1816e-002	1.1334e-005
4	2.808	1.2784e-001	2.1831e-004
5	5.054	3.0954e-003	9.3063e-007
6	2.73	1.1087e-002	9.7014e-006
7	5.007	1.0913e-002	9.2435e-006
8	2.839	6.4753e-002	2.0033e-004
9	0.858	1.8870e-001	2.0992e-004
10	1.466	1.8526e-002	1.0021e-005
11	1.435	7.3802e-003	3.9433e-006
12	4.82	2.0004e-002	3.4435e-005
13	2.808	6.7283e-001	2.6318e-003
14	**5.585**	**2.4106e-003**	**6.7965e-007**
15	3.245	9.0335e-003	6.0896e-006
16	5.554	1.4037e-002	1.0501e-005
17	2.933	2.8708e-001	1.1994e-003
18	0.951	1.5055e-002	2.2534e-005
19	1.295	1.6890e-002	4.8712e-006
20	**1.31**	**6.7920e-003**	**1.5827e-006**
21	14.898	6.8691e-002	6.4766e-005
22	26.692	6.9895e-002	1.1103e-004
23	14.992	3.2526e-002	2.8045e-005
24	30.639	1.2603e-001	2.2272e-004
VF	**3.666**	**1.9453e-001**	**6.0571e-004**

Table B.11: Results for constructing a model of dimension $k = 32$ from a data set obtained from a device with $N_p = 8$ ports

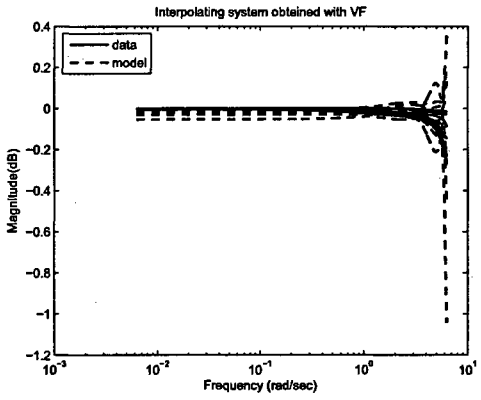
computational times were too large (for example, algorithms 5, 7, 14, 16). Overall, we conclude that algorithms 6 10, 11 provide the best accuracy for a reasonable computational time (less than 4.5s), with 11 giving the best order $k = 40$ model.

We believe that we have already found a good model for the data, so we do not need to search any further. Increasing the order of the desired interpolant would decrease the errors, but will also increase the computational time.

We checked how well the magnitude and the angle of some of the entries are modeled in Figure B.21 for order $k = 40$ models.

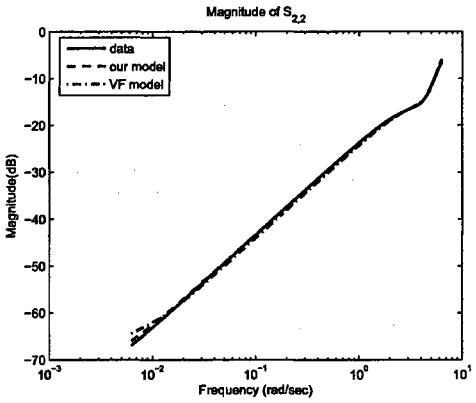


(a) $k = 40$ system with Algorithm 11

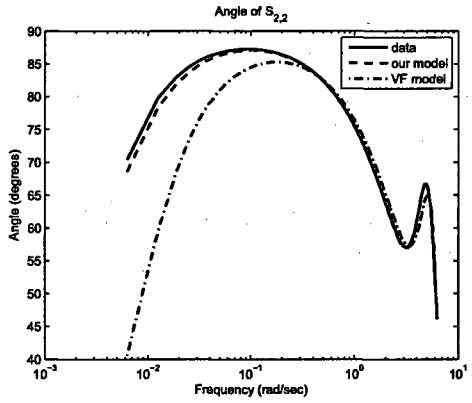


(b) $k = 40$ system with VF

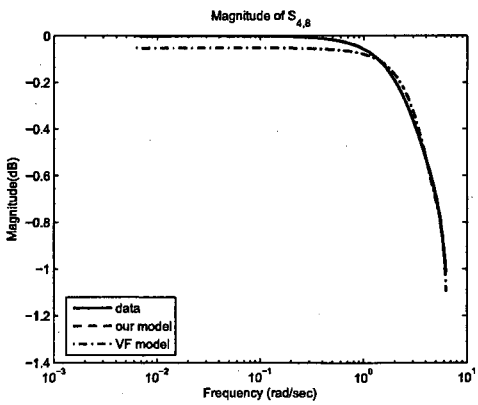
Figure B.20: Models of dimension $k = 40$ obtained with algorithm 11 and with VF for a data set with $N_p = 8$ ports



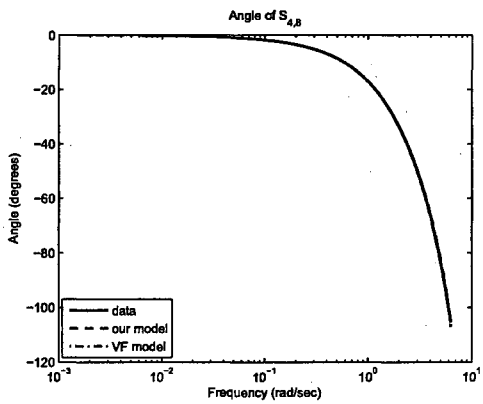
(a) Magnitude of $S_{2,2}$



(b) Angle of $S_{2,2}$



(c) Magnitude of $S_{4,8}$



(d) Angle of $S_{4,8}$

Figure B.21: Comparison of the models built with algorithm 11 and with VF to the data obtained from a device with $N_p = 8$ ports

Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	26.224	1.4526e-002	1.0198e-006
1 with SVD	25.101	4.0619e-002	8.9581e-006
2 with random	25.132	1.9311e-002	3.0046e-006
2 with SVD	17.691	3.8186e-002	1.4036e-005
3	8.58	7.5055e-003	3.4849e-006
4	4.18	3.5321e-002	7.1072e-005
5	7.909	4.5152e-003	5.4614e-007
6	4.43	7.8872e-003	8.0739e-007
7	8.127	3.8784e-003	6.3470e-007
8	4.024	1.2170e-001	3.6540e-005
9	1.279	2.5130e-002	8.4495e-006
10	1.981	4.0182e-003	4.4965e-007
11	**2.012**	**1.5793e-003**	**2.6598e-007**
12	7.846	3.5184e-002	1.2238e-005
13	4.446	3.1522e-002	7.6665e-005
14	8.798	3.8359e-003	2.6966e-007
15	4.024	9.1207e-003	5.1423e-006
16	7.612	1.0421e-002	9.2461e-007
17	4.524	3.0279e-002	2.2630e-005
18	1.357	1.3142e-002	2.9258e-006
19	1.762	2.2939e-002	5.7064e-006
20	1.809	1.5789e-002	1.3231e-006
21	24.352	3.3063e-002	2.0491e-005
22	29.578	1.3183e-001	1.1610e-004
23	23.104	2.5668e-002	2.2246e-005
24	33.119	4.5209e-002	1.4628e-005
VF	**4.072**	**1.0841e-001**	**1.4952e-004**

Table B.12: Results for constructing a model of dimension $k = 40$ from a data set obtained from a device with $N_p = 8$ ports

B.6.5 1000 measurements from a device with 4 ports

This data set contains $N_s = 1000$ frequency samples between $5 \cdot 10^{-3}$MHz and 5MHz. In order to avoid numerical instabilities, all frequencies were scaled by 10^{-6}.

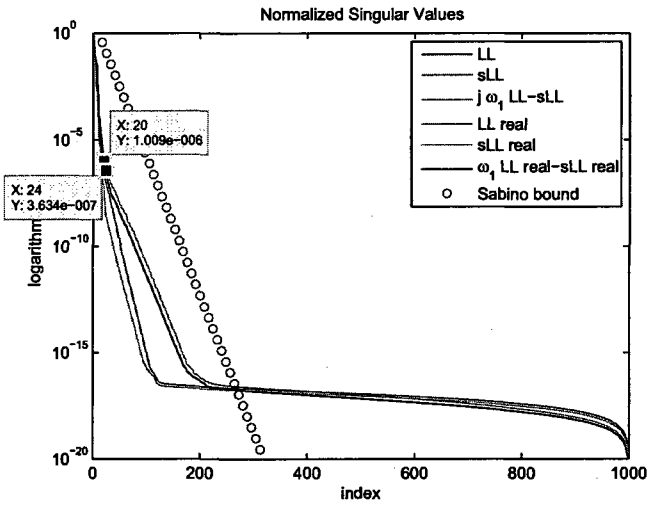


Figure B.22: Singular value drop of the Loewner pencil

First, we construct the big Loewner and shifted Loewner matrices in the real and complex approach using random sampling directions of the left and right tangential interpolation data. Next, we compute how fast the singular values drop and check whether Sabino's bound is able to match the drop of the normalized singular values. Figure B.22 shows that the predicted decay is much slower than the actual decay, even though the decay of the singular values of the pencil built using the real approach are modeled quite closely.

The GUPTRI software takes very long to run on such big matrices (the matrices are of dimension 1000) and we could not obtain an answer even after 2h.

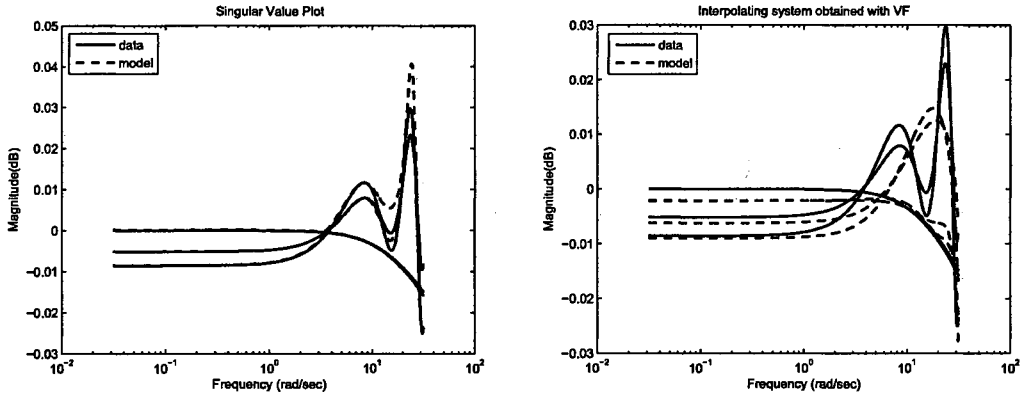
Due to the fact that, to compare models obtained with our algorithms to VF we need to choose the order of the interpolating system as a multiple of the number of ports, we are presenting the results for interpolants of degree $k = 20$ (choosing k less than 20 leads to bad interpolants) in Table B.13 and Figure B.23.

Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	32.433	1.1416e-003	2.2869e-008
1 with SVD	27.643	2.2644e-003	1.0428e-007
2 with random	30.498	4.1158e-003	1.7742e-007
2 with SVD	19.765	8.1488e-003	1.8118e-006
3	2.324	1.1091e-003	1.1473e-007
4	1.294	9.0844e-002	4.9992e-005
5	2.511	6.8085e-004	1.8761e-008
6	1.918	1.2451e-002	1.1117e-005
7	2.62	9.8608e-004	1.0902e-007
8	1.388	2.4043e-002	1.5458e-005
9	0.702	9.7015e-003	1.9920e-006
10	0.92	2.0534e-003	4.7511e-007
11	1.138	3.3258e-003	6.2449e-007
12	2.605	2.5090e-003	5.5838e-007
13	1.544	6.1150e-002	1.5277e-004
14	**2.542**	**4.7609e-004**	**1.8893e-008**
15	1.56	2.1207e-002	9.6297e-006
16	2.184	1.4439e-003	1.8504e-007
17	1.965	4.2290e-002	7.9635e-005
18	0.67	2.8536e-003	1.8233e-007
19	0.889	2.3238e-003	3.5872e-007
20	**0.873**	**1.8205e-003**	**1.9219e-007**
21	7.285	1.8506e-002	5.6719e-006
22	26.723	3.4570e-003	8.6978e-007
23	6.614	3.5646e-003	8.0970e-007
24	32.62	6.0723e-003	1.6875e-006
VF	**1.887**	**4.6749e-003**	**1.2563e-006**

Table B.13: Results for constructing a model of dimension $k = 20$ from a data set obtained from a device with $N_p = 4$ ports

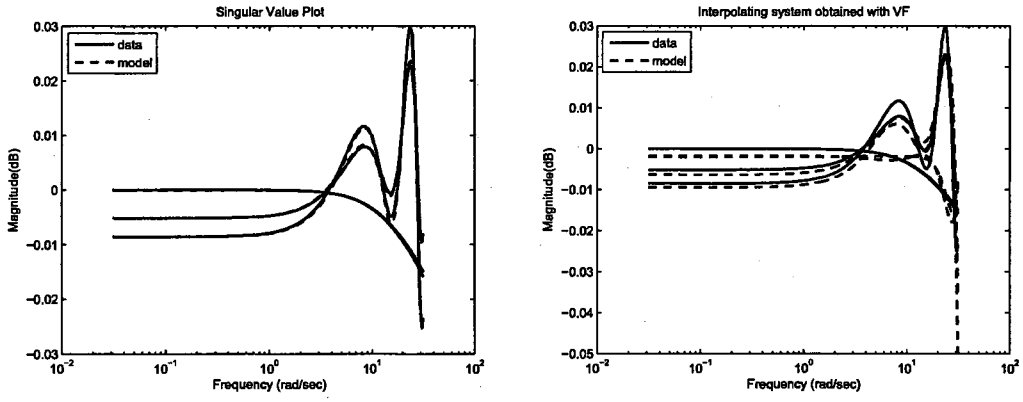
Clearly, even though some of the algorithms proposed yielded good models, the computational times were too large (for example, algorithms 1, 2, 5, 7, 12, 14, 16, 22, 23). The smallest error was obtained with algorithm 14, but the computational time is larger than 2.5s. Overall, we conclude that algorithms 10, 11, 18, 19, 20 provide the best accuracy for a reasonable computational time (less than 1.2s).

Next, let us compare the computational time and the errors when an order $k = 24$ model is generated using all the algorithms. The results are presented in Table B.14 and Figure B.24.



(a) $k = 20$ system with algorithm 20 (b) $k = 20$ system with VF

Figure B.23: Models of dimension $k = 20$ obtained with algorithm 20 and with VF for a data set with $N_p = 4$ ports



(a) $k = 24$ system with algorithm 11 (b) $k = 24$ system with VF

Figure B.24: Models of dimension $k = 24$ obtained with algorithm 11 and with VF for a data set with $N_p = 4$ ports

Clearly, even though some of the algorithms proposed yielded good models, the computational times were too large (for example, algorithms 1, 2, 3, 5, 7, 12, 14, 16 and 22). The smallest error was obtained with algorithm 14, but the computational time is larger than 3s. Overall, we conclude that algorithms 10, 11, 18, 19 and 20 provide the best accuracy for a reasonable computational time (less than 1.7s).

Figures B.23 and B.24 show that the singular values of the S-parameter data matrices exceed the 0dB level, most probably due to measurement errors. Since the largest singular value of the S-parameter matrices in the frequency band of interest is

Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	30.592	1.2711e-003	2.1413e-008
1 with SVD	27.565	1.1299e-002	1.4577e-006
2 with random	29.578	1.5311e-003	2.3169e-008
2 with SVD	20.28	2.8659e-003	2.2142e-007
3	3.79	1.1647e-003	7.4737e-008
4	1.996	1.1120e-002	3.0120e-006
5	3.213	4.4205e-005	1.9771e-010
6	1.622	1.2906e-003	1.8712e-007
7	3.775	2.9419e-004	8.9313e-009
8	1.918	1.9541e-003	3.1849e-007
9	0.904	4.3874e-003	8.9266e-007
10	1.638	1.2025e-003	2.0533e-008
11	**1.17**	**2.8887e-004**	**7.0427e-009**
12	3.01	7.4794e-004	5.0905e-008
13	1.965	8.2669e-003	6.0158e-006
14	**3.322**	**3.7951e-005**	**1.1710e-010**
15	2.23	2.9175e-003	6.6950e-007
16	3.619	3.3897e-003	4.2789e-008
17	2.324	1.7299e-003	2.7676e-007
18	1.076	7.1898e-004	3.3398e-008
19	1.279	9.5565e-004	1.1926e-008
20	1.388	5.6486e-004	9.9789e-009
21	9.703	3.7970e-003	9.6760e-007
22	27.129	1.8681e-003	4.8144e-008
23	9.407	5.4082e-002	1.6626e-005
24	33.431	3.8422e-003	3.4520e-007
VF	**1.856**	**4.4021e-003**	**5.8791e-007**

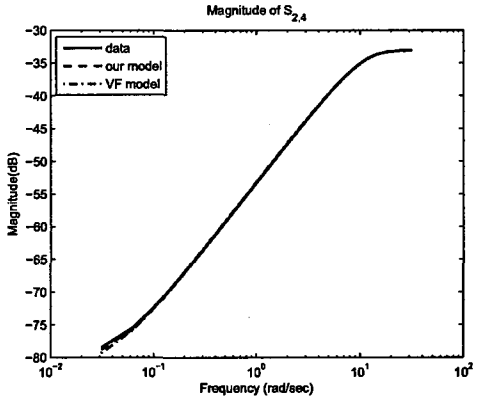
Table B.14: Results for constructing a model of dimension $k = 24$ from a data set obtained from a device with $N_p = 4$ ports

1.0034, we conclude that, in order to interpolate the singular values which are larger than 1, one needs to create non-passive models. Indeed, the order $k = 24$ model built with algorithm 11 has an $\mathcal{H}_\infty$ norm of 1.0013, while the same order macromodel built with vector fitting has an $\mathcal{H}_\infty$ norm of 1.5953. On the other hand, due to the fact that the singular values are all very close to 0dB, we deduce that the underlying device from which the measurements were taken is a loss-free system.

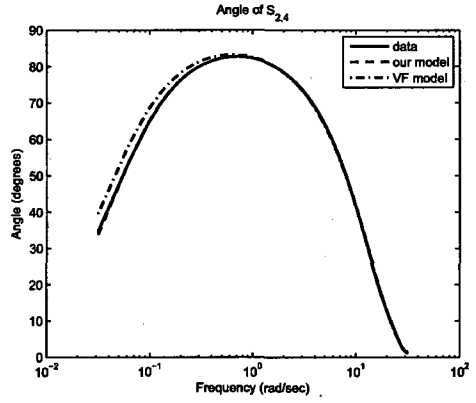
We believe that we have already found a good model for the data, so we do not need to search any further. Increasing the order of the desired interpolant would

decrease the errors, but will also increase the computational time.

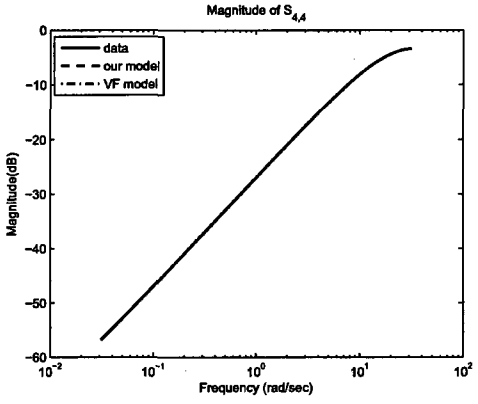
We also checked how well the magnitude and the angle of some of the entries are modeled. Figure B.25 compares the measured $S_{2,4}$ and $S_{4,4}$ entries to the model obtained with algorithm 11 and to the one obtained with VF. Clearly, our model is hardly distinguishable from the data, while VF shows small deviations.



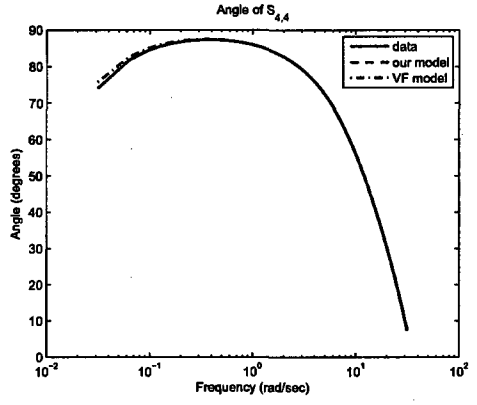
(a) Magnitude of $S_{2,4}$



(b) Angle of $S_{2,4}$



(c) Magnitude of $S_{4,4}$



(d) Angle of $S_{4,4}$

Figure B.25: Comparison of the models built with algorithm 11 and with VF to the data obtained from a device with $N_p = 4$ ports

B.6.6 1000 measurements from a device with 2 ports

This data set contains $N_s = 1000$ frequency samples between 6MHz and 6MHz. In order to avoid numerical instabilities, all frequencies were scaled by 10^{-6}.

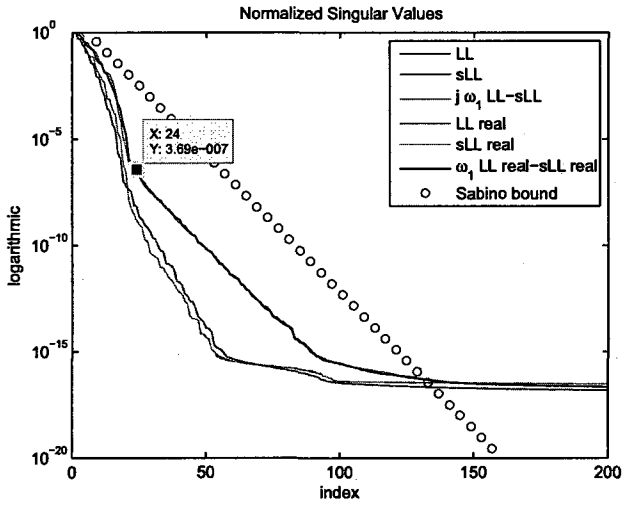


Figure B.26: Singular value drop of the Loewner pencil

First, we construct the big Loewner and shifted Loewner matrices in the real and complex approach using the sampling directions of the left and right tangential interpolation data as outlined in Eq. (5.11), (5.12), (5.22) and (5.23). Next, we compute how fast the singular values drop and check whether Sabino's bound is able to match the drop of the normalized singular values. Figure B.26 shows that the predicted decay is much slower than the actual decay, even though the decay of the singular values of the pencil built using the real approach are modeled quite closely. Recall that the bound cannot be applied to the real approach, so the approximated bound should be compared to the actual drop of the singular values in the complex approach. It predicts that the regular part of the Loewner pair is of dimension 135. Note that the plot only shows the first 200 normalized singular values. The singular values which are not shown are smaller than machine precision.

The GUPTRI software takes very long to run on such big matrices (the matrices

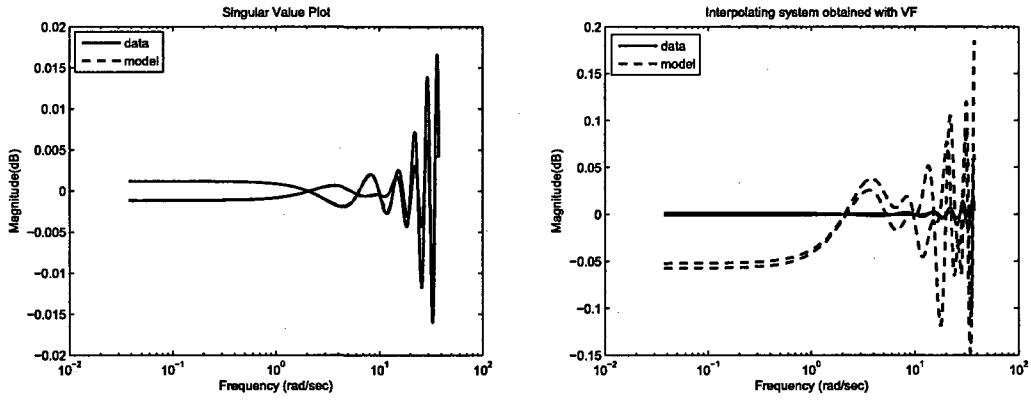
are of dimension 1000) and we could not obtain an answer even after 2h.

Due to the fact that, to compare models obtained with our algorithms to VF we need to choose the order of the interpolating system as a multiple of the number of ports, we are presenting the results for interpolants of degree $k = 28$ (choosing k less than 28 leads to worse interpolants) in Table B.15 and Figure B.27.

Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	26.38	4.3944e-002	4.7972e-005
1 with SVD	30.982	2.6269e-002	6.8061e-006
2 with random	24.96	3.7466e-002	1.3261e-005
2 with SVD	22.963	2.6102e-002	7.4235e-006
3	2.823	1.5189e-002	8.7366e-007
4	1.684	3.6011e-003	7.7193e-007
5	2.886	1.3174e-002	3.1592e-007
6	1.528	1.0611e-002	5.2523e-006
7	2.964	2.9335e-004	5.2892e-009
8	1.575	7.9290e-003	4.7926e-006
9	1.482	6.7939e-003	7.0164e-007
10	1.747	2.8681e-004	6.4070e-009
11	**1.778**	**1.6759e-004**	**1.9140e-009**
12	2.605	2.0318e-003	2.0782e-007
13	1.965	1.0278e-002	7.3202e-006
14	**3.088**	**1.4819e-004**	**8.8631e-010**
15	1.825	1.0743e-002	4.7023e-006
16	2.87	2.9335e-004	5.2892e-009
17	1.606	4.2382e-003	1.6673e-006
18	1.591	3.4887e-003	7.5215e-008
19	1.809	1.2864e-003	8.6758e-008
20	1.684	6.0290e-004	1.4344e-008
21	8.689	7.9227e-002	2.2274e-004
22	24.508	7.1005e-002	8.6299e-005
23	8.19	9.6631e-002	9.7748e-004
24	29.952	3.0177e-002	4.2579e-005
VF	**1.685**	**3.2230e-002**	**7.9527e-005**

Table B.15: Results for constructing a model of dimension $k = 28$ from a data set obtained from a device with $N_p = 2$ ports

Clearly, even though some of the algorithms proposed yielded good models, the computational times were too large (for example, algorithms 7, 14, 16). The smallest error was obtained with algorithm 14, but the computational time is larger than 3s.



(a) $k = 28$ system with algorithm 11 (b) $k = 28$ system with VF

Figure B.27: Models of dimension $k = 28$ obtained with algorithm 11 and with VF for a data set with $N_p = 2$ ports

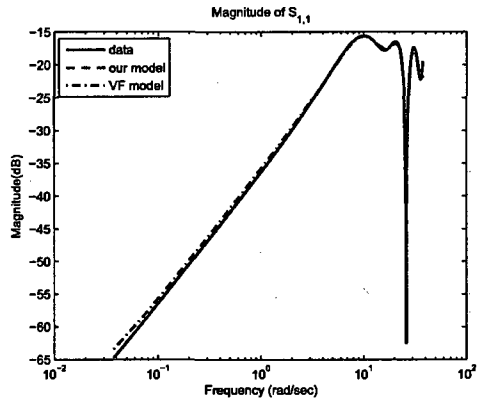
Overall, we conclude that algorithms 10, 11, 18, 19 and 20 provide the best accuracy for a reasonable computational time (less than 1.9s).

Figure B.27 shows that the singular values exceed the 0dB level, most probably due to measurement errors. Since the largest singular value of the S-parameter matrix in the frequency band of interest is 1.0019, we conclude that, in order to interpolate the singular values which are larger than 1, one needs to create non-passive models. Indeed, the order $k = 28$ model built with algorithm 11 has an $\mathcal{H}_\infty$ norm of 1.0017, while the same order macromodel built with vector fitting has an $\mathcal{H}_\infty$ norm of 1.0168. On the other hand, due to the fact that the singular values are all very close to 0dB, we deduce that the underlying device from which the measurements were taken is a loss-free system.

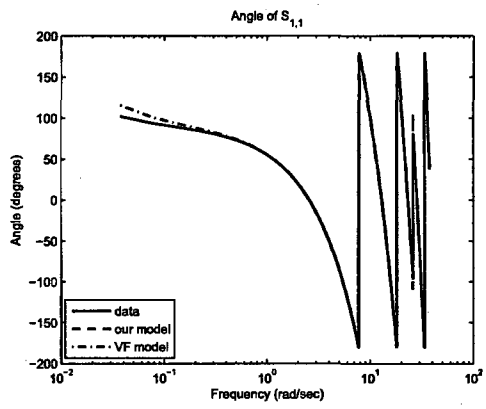
We believe that we have already found a good model for the data, so we do not need to search any further. Increasing the order of the desired interpolant would decrease the errors, but will also increase the computational time.

We also checked how well the magnitude and the angle of the entries are modeled. Figure B.28 compares the measured $S_{1,1}$ and $S_{1,2}$ entries to the model obtained with algorithm 11 and to the one obtained with VF. The underlying device is reciprocal

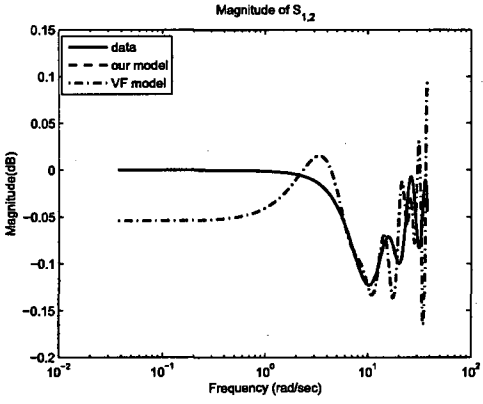
due to the fact that $S_{2,2} = S_{1,1}$ and $S_{1,2} = S_{2,1}$, so we omitted the plots of $S_{2,2}$ and $S_{2,1}$. Clearly, our model is hardly distinguishable from the data, while VF shows deviations.



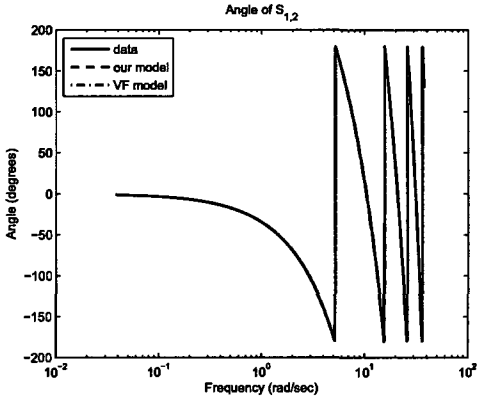
(a) Magnitude of $S_{1,1}$



(b) Angle of $S_{1,1}$



(c) Magnitude of $S_{1,2}$



(d) Angle of $S_{1,2}$

Figure B.28: Comparison of the models built with algorithm 11 and with VF to the data obtained from a device with $N_p = 2$ ports

B.6.7 201 measurements from a device with 2 ports

This data set contains $N_s = 201$ frequency samples between 50MHz and 40GHz.

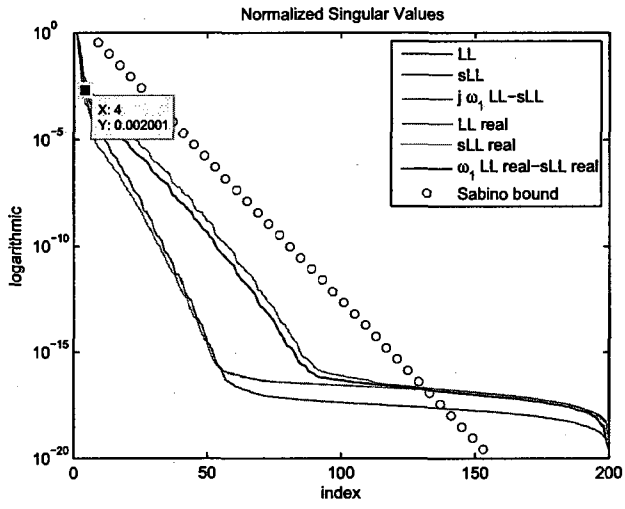


Figure B.29: Singular value drop of the Loewner pencil

First, we construct the big Loewner and shifted Loewner matrices in the real and complex approach using the sampling directions of the left and right tangential interpolation data as outlined in Eq. (5.11), (5.12), (5.22) and (5.23). Next, we compute how fast the singular values drop and check whether Sabino's bound is able to match the drop of the normalized singular values. Figure B.29 shows that the predicted decay is much slower than the actual decay, even though the decay of the singular values of the pencil built using the real approach are modeled quite closely. Recall that the bound cannot be applied to the real approach, so the approximated bound should be compared to the actual drop of the singular values in the complex approach. It predicts that the regular part of the Loewner pair is of dimension 130. We notice a steep decay in the first 4 singular values, so we will compute models of order $k = 4$.

The matrix dimensions are 201 for the complex approach and 200 for the real approach, respectively.

Applying the GUPTRI software to the Loewner pencil constructed using the complex approach reveals a singular part of dimension $198-0 = 198 = 199-1$, one infinite eigenvalue $(1-0=1)$ and a finite part of dimension 2 for a value of EPSU, the uncertainty in the data, of 10^{-5}. The data is given with 11 digits after the decimal point, but the singular value plot shows that there is a lot of noise presentin the data. Any other input value for the uncertainty in the data leads to no finite part. The poles given by GUPTRI are recovered as the eigenvalues of the top right 3×3 matrix sub-pencil.

```
>> [Sc,Tc,Pc,Qc,kstrc]=guptri(sLL,LL,10^(-5));
>> kstrc
kstrc =
   198    -1   199     0    -1     2    -1
     0    -1     1     0    -1     2    -1
>> eig(Sc(1:3,199:201),Tc(1:3,199:201))
ans =
     -9.739102893295235e+09 - 2.932383863969493e+07i
     -1.086118287564900e+12 + 1.697843024203552e+10i
                        Inf
```

When we construct the matrices in the real approach and use a value of the uncertainty in the data of 10^{-7}, we obtain a right singular part of dimension $190-5 = 185$, 4 zero eigenvalues with multiplicity 1 $(5-1=4)$, no zero eigenvalues with multiplicity 2 $(1-1=0)$, a Jordan block of dimension 3 $(1-0=1)$ associated with the zero eigenvalue, a left singular part of dimension $185-2 = 183$, no infinite eigenvalues $(2-2=0,\ 1-1=0,\ 1-1=0,\ 1-1=0,\ 1-1=0)$ and a regular part of order 2.

```
>> [Sr,Tr,Pr,Qr,kstrr]=guptri(sLLN,LLN,10^(-7));
kstrr =
  Columns 1 through 11
   190     1     1     0    -1   185     2     1     1     1     1
     5     1     1     0    -1     2     1     1     1     1     0
  Columns 12 through 14
    -1     2    -1
    -1     2    -1
```

This means that the first 185 columns and the last 183 rows of the matrices $\mathbf{S}_c$ and $\mathbf{T}_c$ are zero, while the top right 15×15 sub-pencil has the following eigenvalues:

```
>> eig(Sr(1:15,186:200),Tr(1:15,186:200))
ans =
                            0
                            0
                            0
                            0
                            0
                            0
                            0
   -4.482540496224155e+08 + 1.207610836802324e+11i
   -4.482540496224170e+08 - 1.207610836802324e+11i
   -1.513449880900041e+09 - 1.288626766397430e+11i
   -2.167131060445620e+10 - 1.571635437154970e+11i
   -1.513449880899482e+09 + 1.288626766397425e+11i
   -2.167131060444972e+10 + 1.571635437154978e+11i
   -1.037717834071183e+12
   -8.534909249932325e+12
```

We have recovered 7 zero eigenvalues, as expected, but it is unclear how to extract the finite part.

This example was analyzed in [25], where a 4-th order model was constructed using vector fitting. The drop in the normalized singular values associated with the Loewner pencil also indicated that the order of the underlying system is 4. Let us compare the models obtained with our algorithms to the one obtained with VF. We are presenting the results for interpolants of degree $k = 4$ in Table B.16 and Figure B.30.

Clearly, all the algorithms proposed required less time to generate a 4-th order model than VF did. Except for algorithms 3 and 4, all the remaining ones yielded better models than the one produced by VF. The smallest error was obtained with algorithm 1.

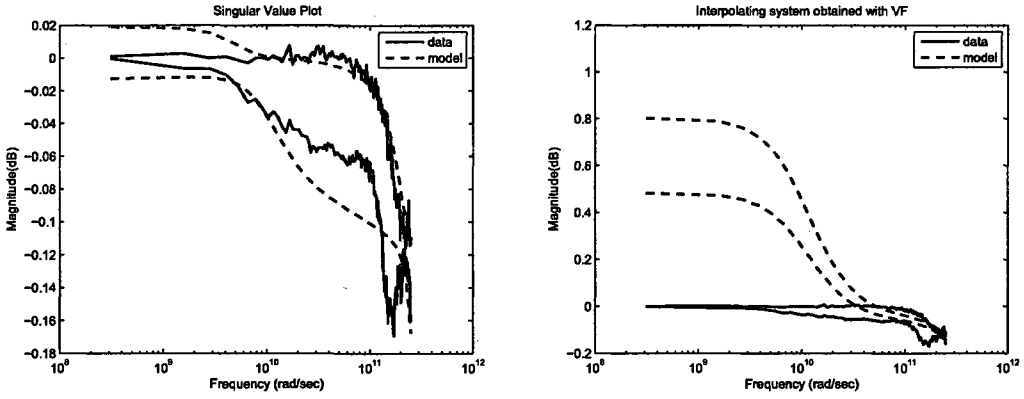
Figure B.27 shows that the singular values exceed the 0dB level, most probably due to measurement errors. Since the largest singular value of the S-parameter matrix

Algorithm	CPU time (s)	$\mathcal{H}_\infty$	$\mathcal{H}_2$
1 with random	0.109	8.2005e-003	1.7175e-005
1 with SVD	**0.28**	**8.4701e-003**	**1.6279e-005**
2 with random	0.249	3.6905e-002	4.0569e-005
2 with SVD	0.156	1.6998e-002	5.6383e-005
3	0.124	4.5806e-001	4.9218e-003
4	0.078	7.5772e-001	2.5819e-002
5	0.062	1.4314e-002	5.9268e-005
6	0.124	4.3936e-002	1.3824e-004
7	0.093	5.7292e-002	2.1666e-004
8	0.093	6.4476e-002	6.0989e-004
9	0.093	5.6725e-002	6.3479e-004
10	0.14	1.6004e-002	4.8683e-005
11	0.078	1.7238e-002	7.7507e-005
12	0.078	1.6940e-002	6.7017e-005
13	0.093	1.6195e-001	3.4930e-003
14	0.078	1.4501e-002	6.0657e-005
15	0.109	4.3936e-002	1.3824e-004
16	0.062	5.8145e-002	2.2400e-004
17	0.156	6.4476e-002	6.0989e-004
18	0.109	5.8017e-002	6.6194e-004
19	0.109	2.0053e-002	8.7373e-005
20	0.046	1.7238e-002	7.7507e-005
21	0.234	4.0008e-002	2.4591e-004
22	0.171	1.9642e-002	6.5533e-005
23	0.062	2.1842e-002	1.3559e-004
24	0.093	1.1340e-001	8.6714e-004
VF	**0.374**	**1.6923e-001**	**3.9597e-003**

Table B.16: Results for constructing a model of dimension $k = 4$ from a data set obtained from a device with $N_p = 2$ ports

in the frequency band of interest is 1.0009, we conclude that, in order to interpolate the singular values which are larger than 1, one needs to create non-passive models. Indeed, the order $k = 4$ model built with algorithm 1 has an $\mathcal{H}_\infty$ norm of 1.0022, while the same order macromodel built with vector fitting has an $\mathcal{H}_\infty$ norm of 1.0968. On the other hand, due to the fact that the singular values are all very close to 0dB, we deduce that the underlying device from which the measurements were taken is a loss-free system.

We also checked how well the magnitude and the angle of the entries are modeled.

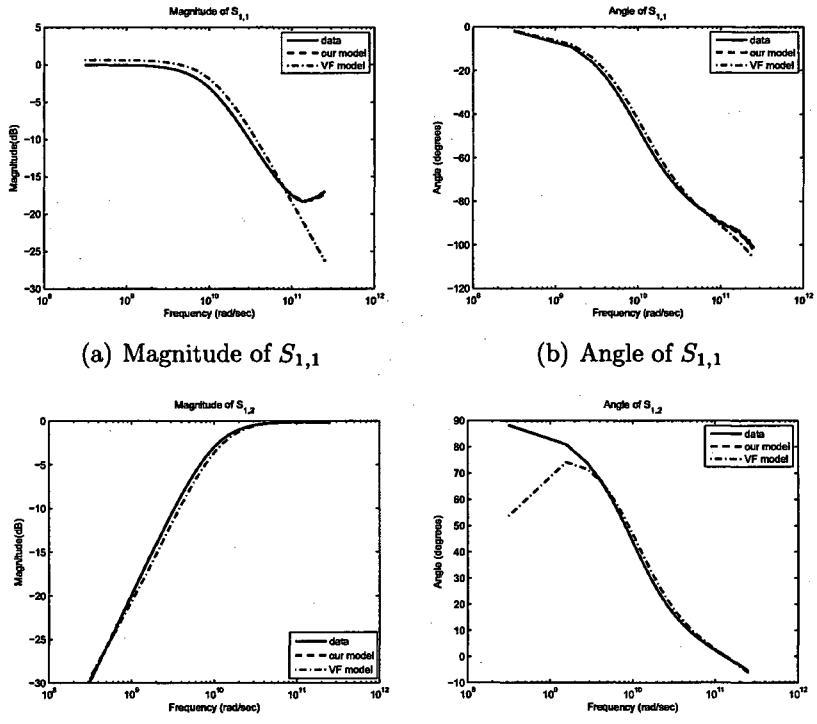


(a) $k = 4$ system with algorithm 1

(b) $k = 4$ system with VF

Figure B.30: Models of dimension $k = 4$ obtained with algorithm 1 and with VF for a data set with $N_p = 2$ ports

Figure B.31 compares the measured $S_{1,1}$ and $S_{1,2}$ entries to the model obtained with algorithm 1 and to the one obtained with VF. The underlying device is reciprocal due to the fact that $S_{2,2} = S_{1,1}$ and $S_{1,2} = S_{2,1}$, so we omitted the plots of $S_{2,2}$ and $S_{2,1}$. Clearly, our model is hardly distinguishable from the data, while VF shows deviations.



(a) Magnitude of $S_{1,1}$

(b) Angle of $S_{1,1}$

(c) Magnitude of $S_{1,2}$

(d) Angle of $S_{1,2}$

Figure B.31: Comparison of the models built with algorithm 1 and with VF to the data obtained from a device with $N_p = 2$ ports